DUPLICATE

Forecasting Research

Forecasting Research Division
Technical Report No. 129

The Application of Pattern Recognition Techniques To Cloud Detection

by

GEORGE PANKIEWICZ

November 1994

METEOROLOGICAL OFFICE.

Forecasting Research Division Technical Report No.129

The application of pattern recognition techniques to cloud detection.

Bracknell::::1994:Pp.i+74:30cm:

94110691 551.501.776
551.501.86

ORGS UKMO F

National Meteorological Library
FitzRoy Road, Exeter, Devon. EX1 3PB

Meteorological Office
London Road
Bracknell
Berkshire
RG12 2SZ
United Kingdom

The Application Of Pattern Recognition Techniques To Cloud Detection

Forecasting Research Division Technical Report No. 129

GEORGE PANKIEWICZ

FR Division, Meteorological Office, London Road, Bracknell, Berkshire, RG12 2SZ, UK

© Crown Copyright November 1994

Contents

1 Executive Summary 1

2 Introduction 2

3 Pattern Recognition Fundamentals 3

4 Constructing The Feature Space 6
- 4.1 Spectral Features . . . 7
- 4.2 Textural Features . . . 8
- 4.3 Spatial Features . . . 13
- 4.4 Contextual Features . . . 16
- 4.5 Feature Selection and Extraction . . . 18

5 Pattern Classification 22
- 5.1 Supervised Classical Techniques . . . 22
- 5.2 Unsupervised Classical Techniques . . . 29
- 5.3 Introduction to Artificial Neural Networks . . . 34
- 5.4 Supervised Multilayer Perceptron Techniques . . . 38
- 5.5 Unsupervised Kohonen Techniques . . . 44

6 Scale Context And Image Segmentation 46

7 Cloud Detection And Other Applications 51

8 Conclusions 64
- 8.1 Summary . . . 64
- 8.2 Recommendations for Research . . . 67

9 References 69

1 Executive Summary

The uses of pattern recognition techniques, applied to the detection of clouds and cloud systems in meteorological satellite imagery are discussed. Emphasis is placed on the detection of cloud over a wide range of scales, encompassing subpixel cloud through to synoptic scale systems.

The key elements of pattern recognition techniques and concepts are introduced before describing those which are applicable to cloud recognition. Two stages of the pattern recognition process are discussed in detail, the first relating to the feature vector elements that are the components of the space in which the patterns are formed, and the second concerning the partitioning of that feature space into classes or patterns, depending on whether they are known *a priori* (supervised), or not. All features relevant to the recognition of cloud are examined, and include spectral, textural, spatial and contextual characteristics. Methods of minimising the number of features used, in an effort to increase computational efficiency, are discussed in terms of feature selection and extraction.

Many well-known pattern classification techniques are considered and assessed, including nearest neighbour methods, parallelepiped classifiers, discriminant functions, minimum-distance-to-means procedures and Gaussian maximum likelihood classifiers. Under the heading of unsupervised methods, both hierarchical and partitional techniques are examined, as well as a range of criterion functions for implementing partitional methods. Since 1990, neural networks have been increasingly used as pattern recognition tools in cloud classification studies. Perceptrons, multilayer perceptrons and Kohonen networks are discussed in terms of discriminant analysis, and some of the architectures used in cloud studies are examined.

Cloud classifications performed at a particular scale generally use the pattern recognition techniques described above, but when a range of scales is to be considered, image segmentation and scale context become important. Recently, bottom-up and top-down approaches have been used (in the sense of a change of scale and different levels of classification), but both appear to have shortcomings. A new method of bidirectional scale analysis (BSA) is introduced which employs a combination of these techniques with feedback, and which may overcome the existing limitations.

The major past applications of pattern recognition to cloud identification are considered, including cloud classification for operational or climatological reasons, cloud system classification and objective synoptic analysis for forecasters and for numerical weather prediction (NWP) models, automatic determination of cloud motion winds, precipitation estimation, analysis of cloud physical parameters and satellite image data reduction. A few of the key results which have arisen from cloud classification studies alone in the last 20 years have included an improvement in automatic cloud classification accuracies of over 20% (largely as a result of using textural features and neural networks), the discovery that a spatial resolution no greater than about 250m is required to extract all the information relevant to cloud classification, and the first steps to obtaining a polar stratospheric cloud climatology.

Suggestions for further research at the UK Meteorological Office are given, including cloud system recognition to provide information based on conceptual models for assimilation into NWP, precipitation estimation and cloud climatological studies.

2 Introduction

This technical report is based on the findings of a literature review conducted for the DEV014 project, which was initiated in December 1993 by the Satellite Image Applications Group of FR Division. The aim of DEV014 is to develop a scheme for the detection and identification of clouds and cloud systems from meteorological satellite imagery, using pattern recognition techniques.

The way in which such a scheme might be developed has been left deliberately open, in order to explore as many of the techniques as possible. The emphasis however, is on the identification of both clouds and cloud systems, and this would appear to warrant some form of hierarchical approach, able to differentiate between individual cloud elements on scales of a few kilometres or less, to synoptic systems on scales of a few hundred kilometres.

The report begins by describing a variety of general pattern recognition techniques of relevance to meteorological satellite imagery and cloud detection. (Every pattern recognition problem "acquires" a collection of procedures suited to its special characteristics). However, before looking at each one in detail, section 3 is used to outline the basic ideas and concepts required to understand the route through most of the pattern recognition systems discussed in the literature. This section should be read by anyone unfamiliar with the fundamentals of pattern recognition.

Characteristics of objects in images that typify or distinguish those objects from others are referred to as "features" in pattern recognition, and these are presented in section 4. There are many types of feature, which can be ascribed to one of four main groups: spectral, textural, spatial or contextual. These are all discussed individually, with particular reference to their past use in cloud classification schemes. The first major task in pattern recognition concerns the choice of these features, and the selection (or extraction) of the optimal set (§4.5). Once the feature set has been chosen, a d-dimensional feature space can be created (d representing the number of individual features), which best characterises the relationships between the classes, and becomes the basis for the recognition of those classes.

The next stage of a typical pattern recognition process is to assign classes according to their positions in an appropriately partitioned feature space, and this is discussed in section 5. Many classification schemes have been identified in the past (see for example, the textbook by Duda & Hart (1973)), but this report will concentrate on four main categories. The first (and the most commonly used) category comes under the heading of Supervised Classical Techniques (§5.1), and is immediately followed (§5.2) by a discussion of Unsupervised Classical Techniques, the difference being whether the classes used in training the system are labelled or not.

Since Rumelhart, McClelland and The PDP Research Group published their book "Parallel Distributed Processing" in 1986, there has been an explosion of interest in the use of artificial neural networks for the purpose of pattern recognition (hence the use of the word "Classical" in §5.1 and §5.2). The subsection entitled "Introduction to Artificial Neural Networks" (§5.3) breaks away slightly from the main text, but gives the reader who is unfamiliar with such techniques a chance to understand the essentials. It attempts to explain why neural networks are considered to be of such use, how they function, and gives an example of the basic neural network,

the single layer perceptron.

The more advanced multilayer perceptron is discussed in §5.4, which uses the "backpropagation rule" devised by Rumelhart *et al.* (1986b). This network requires labelled training data, and therefore comes under the title of Supervised Multilayer Perceptron Techniques. Kohonen topological maps are another type of neural network, but are rather different from layered perceptrons. They are used to discover relationships between features for unlabelled training data, and are discussed in §5.5.

Normally, the classification of an image at a particular spatial scale might be the final result of a pattern recognition exercise. However, in the analysis of clouds and cloud systems, it is important to determine the scale over which a particular pattern should dominate, and to put that classification into context. For example, a satellite based detector able to resolve down to 200m over a region of cumulus cloud might observe a collection of pixels containing a large degree of variance in albedo. In contrast, a detector with the ability to resolve down to only 1km would detect an averaged value of those grey-levels in each pixel, and a group of pixels covering several square kilometres might show even less structure and more homogeneity. If however, the scale of analysis for this detector is increased to the order of 100km, new structure is likely to become apparent. It is therefore important to recognise (and set into context) the scale of cloud structure from pixel dimensions right through to synoptic scales. The next stage of a cloud analysis must perform some type of image segmentation with the classification already made, set it into context in an intelligent way, and re-evaluate the image (or part of the image) in terms of other scales. These aspects are all discussed in section 6.

Section 7 attempts to put the various methods into perspective, and give a flavour of the types of application relating to clouds for which pattern recognition techniques have been used. The main area of active research has been cloud classification for either operational or climatological reasons, and the techniques developed in this field are discussed in a historical fashion, with examples of specific applications. The latter part of the section looks at other applications including synoptic scale feature identification for numerical weather prediction improvement, construction of expert based forecasting systems, cloud wind motion studies, precipitation estimation, basic physical process studies and data reduction. The major themes and results of the report are summarised at the beginning of section 8, and a few potential schemes for detecting clouds and cloud systems are then outlined, with a view to recommendations for future research at the UK Meteorological Office.

3 Pattern Recognition Fundamentals

"Paradoxically, we are all expert at perception, but none of us knows much about it", (Duda & Hart, 1973). Human perception is much more complex than one thinks. For example, in reading this paragraph, most of the information that is absorbed is presented in the form of different patterns, occurring as strings of letters. Before even considering the numerous cognitive issues of language processing (for instance, what makes it possible to understand "The boy the man chased kissed the girl"?), the visual system must first solve the pattern recognition problem.

If a computer were set up to do this task, the complexity would become even more apparent. First, image data of the paragraph in a form suitable for computer processing must be obtained (for example, using a camera and digitiser). The computer system would perhaps need to read each letter into a fixed frame, judge the scale, orientation, intensity, and compare the patterns in this frame to a template of all the possible letters. This is an example of a basic pattern recognition process, and it can be identified in terms of three processes:

1. **The transducer stage**
2. **Feature selection or extraction**
3. **Classification**

The process described above, of obtaining image data of a paragraph of text, and converting it into a suitable digital format, corresponds to the transducer stage. It is perhaps worth considering for a moment, the equivalent to this process in the field of meteorological satellite imagery, since later sections of this work will concentrate on the other two aspects of pattern recognition.

Meteorological satellites use scanning radiometers working predominantly in visible or infrared wavebands (and increasingly at microwave régimes, although such data will not be considered in this report). Digital grey-level data from sensors (for example, on board currently operational satellites Meteosat-5 (geostationary) and NOAA-11 and -12 (polar orbiters)) are telemetred to ground receivers, and the strength of each radiance measurement is allocated to a pixel governed by the area over which the measurement was made. The UK Meteorological Office uses a ground station at Lasham (Hampshire), where an automatic processing system, known as Autosat-2 (see for example, Allam *et al.* (1990)) reprojects the image onto a polar stereographic projection, and converts instrument counts into a range of products, which include visible albedos and infrared brightness temperatures.

Going back to the letter recognition example, the next stage is to identify properties of letters in the image that distinguish them from others. Such properties are referred to as *features*, and play a very important rôle in pattern recognition. For example, the intensity levels alone might not vary greatly (black and white only), so other features, including spatial information (the number of straight vertical edges perhaps) or contextual information (is there a dot over a straight bar?) will be very useful in identifying the letters uniquely.

Once suitable features have been identified and their values obtained, each letter will have a particular set of features attached to it, called a *feature vector*. The dimension of the feature vector (usually denoted by d) is just the number of different features used. Features can be thought of as axes in an d-dimensional *feature space*. Each letter then has a particular feature vector associated with it, occupying a unique position in the feature space (see figure 1).

It may turn out that some features become redundant because they provide nearly the same piece of information about each letter (this would occur if all the letters occupied the same position along one of the axes in the feature space). Another possibility is that some other feature is better able to identify certain characteristics.

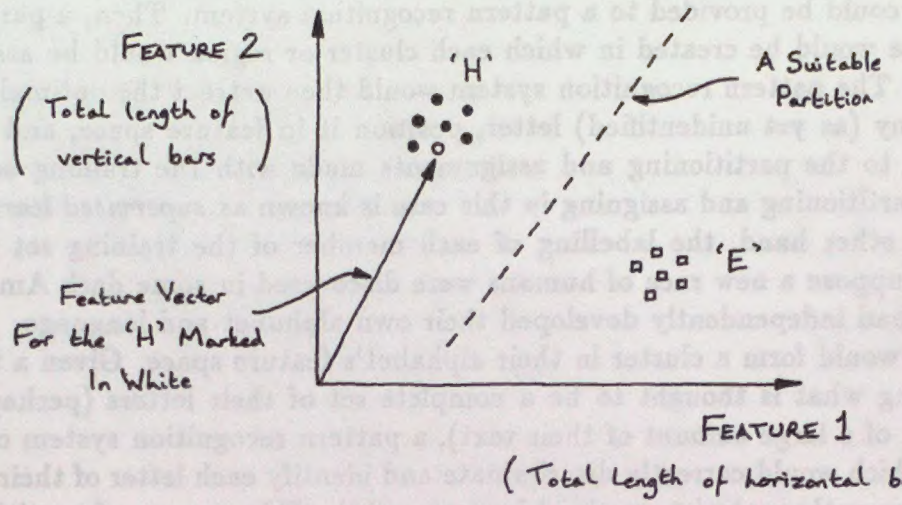

Figure 1: *An example of the positioning of two letters (H and E) in a 2-dimensional feature space. The features chosen measure the total length of all horizontal or vertical bars in the letters, and clusters are produced for a small training set. A partition can be drawn between the clusters which discriminates any new letter as an H or an E.*

The purpose of a *feature extractor* is to reduce the data (and also the amount of computation) by assessing how well the features are separated in feature space (in the example, short straight bars would be separated further from dots than from long straight bars). Other names for the feature extractor also include the receptor, property filter, attribute detector, or preprocessor. A discussion of features common to meteorological satellite imagery, and techniques for feature extraction and selection (which by strict definition, are different) will be presented in detail in section 4.

For any individual letter in the example, there now exists a new optimal feature vector, which has some particular length and direction in feature space. If a series of letters, picked at random from some text, were "positioned" in feature space, a number of clusters would emerge, each one corresponding to a different letter (or class). (The size of the cluster might depend on such things as the ability of the printer to produce identical letters, and the resolving power of the sensing instrument).

The next stage in the pattern recognition process is *classification*. So far, different letters appear to correspond to different clusters in feature space. Classification is nothing more than the task of partitioning up the feature space according to the (statistical) details of the clusters, and identifying which letter, or class, belongs to which cluster. However, there may seem to be a paradox here. How does each cluster member in the feature space (representing a class that the system is trying to identify) become assigned to a given class, when the pattern recognition system does not know what the class is? The answer is that a *training set* of data are required to teach the system which clusters belong to which classes.

At this point, it is sensible to consider whether the classes in the training set are labelled. This may seem trivial, but in fact, it is not. In some circumstances,

the labelling of individual letters from a piece of text are known, and a labelled training set could be provided to a pattern recognition system. Then, a partitioned feature space would be created in which each cluster or region would be assigned a given letter. The pattern recognition system would then extract the optimal feature vector for any (as yet unidentified) letter, position it in feature space, and classify it according to the partitioning and assignments made with the training set. The process of partitioning and assigning in this case is known as *supervised learning.*

On the other hand, the labelling of each member of the training set may be unknown. Suppose a new race of humans were discovered in some dark Amazonian forest, who had independently developed their own alphabet and language. Each of their letters would form a cluster in their alphabet's feature space. Given a training set containing what is thought to be a complete set of their letters (perhaps from photographs of a large amount of their text), a pattern recognition system could be developed which would correctly discriminate and identify each letter of their system (assuming the anthropologist survived long enough!). This process of partitioning a training set without any labels is known as *unsupervised* learning. From the example, it is clear that one of the most important aspects of any unsupervised learning process is the presence of a large and complete set of training data.

So, once a pattern recognition system has been designed and taught, using a significantly large enough set of training data, it is simply a matter of extracting the optimal feature vector from any new unidentified pixel[1], and classifying it according to the partitioning obtained in training. There are of course a number of questions left to answer, such as: how is the optimal feature vector obtained, what constitutes a significantly large training set and how are the training set feature vectors partitioned? These questions will hopefully be answered in the following sections.

4 Constructing The Feature Space

In designing a pattern recognition system for cloud detection, it is not always clear exactly which features should be used to characterise the image data. Although simple grey-level intensities may on the surface appear to hold much of the information in the image, human experience has shown that there are many other characteristics worth considering, including texture, size, shape, context and the presence of shadows (Bunting & Hardy, 1984). These features and others, which will become the basis for a feature space aimed at cloud detection, will now be discussed in detail, whilst a number of methods for selecting the optimal set of such features to use for the purpose of classification will be described later.

In the past, the majority of work concerning pattern recognition of clouds was purely for classification purposes, and the end product would be a thematic map of individual samples taken from satellite imagery data. Nearly all workers used spectral and textural features only, and good examples of cloud classification schemes of this genre were described by Parikh (1977), Harris & Barrett (1978), Ebert (1987), Garand (1988) and Bankert (1994). However, a few alternative types of feature were

[1]Strictly, the entities to be recognised should be referred to as *samples*, since classification might be required over a range of scales, perhaps using neighbourhood information (see §4.4).

also investigated (described later), which might well yield interesting results when changes of scale from clouds to cloud systems are taken into account.

It seems appropriate to list the various features for cloud detection under 4 general headings, although it should be remembered that these characteristics are very much interrelated:

1. **Spectral** - intensities and multispectral characteristics.
2. **Textural** - statistical measures of the local neighbourhood (including spatial frequencies and fractal dimensions).
3. **Spatial** - size and shape.
4. **Contextual** - relationships with other cloud systems.

4.1 Spectral Features

Perhaps the most obvious features of satellite imagery are spectral or multispectral grey-level intensities, and some authors have relied solely on this type of measurement in the past, usually with the emphasis on investigating individual cloud cover parameters via some form of radiative transfer model (Arking & Childs, 1985; Rossow *et al.*, 1985; Saunders & Kriebel, 1988). These parameters included cloud fraction, optical thickness, cloud-top temperature, liquid water content and microphysical model indices, the last used to represent physical properties of individual cloud particles.

Perhaps this is a good point to emphasise that it is not the aim of this work to make sole use of such radiative models, which necessarily build on a number of assumptions regarding the way in which atmospheric, surface and cloud properties affect satellite measurements. The approach is to look for patterns resulting from relationships between a large range of features that clouds exhibit, and to be able to later identify cloud types and systems from those patterns. Some cloud classification schemes have used spectral features alone (Desbois *et al.*, 1982; Farki *et al.*, 1992), but they are usually cases where the feature space is analysed by unsupervised clustering techniques (described later in §5.2).

Visible and infrared grey-levels can be particularly useful in discriminating cloud type with height, and this fact was put to use in the detailed study of Garand (1988). This is because in visible wavebands, brighter clouds are more reflective, whilst in the infrared, lower clouds are warmer. Therefore low water clouds such as cumulus, stratocumulus and stratus, tend to appear warmer and brighter than high ice clouds such as cirrus, cirrostratus and cumulonimbus (Bunting & Hardy, 1984). In Garand's study, four features relating to height were computed based on the cloud top temperature (θ_t), originally obtained from the infrared brightness temperature corresponding to the 97th percentile in an infrared histogram. The effective cloud top height η(km) was then calculated using the relation:

$$\eta = \frac{(\theta_s - \theta_t)}{6.5} \qquad (1)$$

where θ_s is the sea-surface temperature (obtained from monthly climatology to the nearest 1°C), and a vertical temperature lapse rate of 6.5°Ckm^{-1} is assumed. Garand

did point out that the temperature lapse rate for boundary layer cloud fields is often closer to $10°\text{Ckm}^{-1}$ (dry adiabatic), but as a classification feature, η *was not required to represent a true height.* The other three features represented the relative amounts of cloud at low, middle and high altitudes, and were defined by the percentages of cloudy pixels in 3 different temperature bands based on equation [1] with boundary altitudes at 2 and 6km.

Besides individual grey-levels, further spectral features can be gained by considering a variety of statistics taken over a selection of pixels (for example, the 32 by 32 pixel boxes of the NOAA-7 AVHRR data used by Ebert (1987)). That study included statistical features such as the mean, standard deviation, maximum, minimum, the ratio of maximum to minimum and the range of the grey-levels of all the pixels in the box. In this way, a large number of additional spectral features were calculated (18 more, for 3 AVHRR channels), although it should be remembered that their effectiveness in terms of resolution decreases as the significance of the statistics (number of pixels) increases.

A slightly different use of statistical spectral features was made by Delgado *et al.* (1992). They used the standard deviation over boxes of 9 pixels to search for regions unrepresentative of classes (having a large variance), which they then discarded in order to improve the classification.

4.2 Textural Features

Two fundamentally different methods of textural feature analysis were defined by Haralick (1986): *statistical* texture analysis generates parameters which characterise the stochastic properties of the spatial distribution of grey-levels in an image, whilst in the *structural* approach, scenes are analysed in terms of the organisation and relationships among its substructures. In this work, textural features refer only to those obtained in the statistical sense, and Haralick's structural textural features are renamed *contextual* features, and are discussed later in this section.

It is also important to clarify the difference between the statistical spectral features discussed earlier and the textural features described here. Statistical spectral features, such as the mean of a number of grey-levels in a box, do not characterise the *spatial distribution* of those grey-levels, whereas the textural features described below are fundamentally concerned with the relative positions of grey-levels in particular neighbourhoods.

Haralick (1986) listed the following set of statistical textural measures: grey-level co-occurrence, gradient analysis, orthogonal transformation, autocorrelation, mathematical morphology of tonal primitives and relative extrema density. However, most cloud classification techniques have relied on statistics from only the first four of these areas, with a variant of grey-level co-occurrence being by far the most popular.

The general co-occurrence $P(g_1, g_2)$ of grey-levels g_1 and g_2 for an image is defined as the number of pairs of pixels having grey-levels g_1 and g_2, which are in a fixed spatial relationship, (such as a fixed distance apart or a fixed distance and a fixed angle). The co-occurrence matrix is usually normalised by dividing each entry by the sum of all entries in the matrix. Statistics from such co-occurrence matrices have been used to analyse textures in a variety of satellite images, and have included contrast,

angular second moment (sometimes referred to as energy), entropy, homogeneity and correlation (Haralick, 1986; Welch *et al.*, 1988; Lee *et al.*, 1992). Using Lee *et al.*'s form for correlation, and a grey-level range of G (256 for an 8-bit data channel), their statistics are defined as:

$$\text{Contrast} = \sum_{g_1=1}^{G}\sum_{g_2=1}^{G} P(g_1,g_2)\cdot(g_1-g_2)^2 \tag{2}$$

$$\text{ASM} = \sum_{g_1=1}^{G}\sum_{g_2=1}^{G} P^2(g_1,g_2) \tag{3}$$

$$\text{Entropy} = -\sum_{g_1=1}^{G}\sum_{g_2=1}^{G} P(g_1,g_2)\cdot log P(g_1,g_2) \tag{4}$$

$$\text{Homogeneity} = \sum_{g_1=1}^{G}\sum_{g_2=1}^{G} \frac{P(g_1,g_2)}{1+|g_1-g_2|} \tag{5}$$

$$\text{Correlation} = \sum_{g_1=1}^{G}\sum_{g_2=1}^{G} \frac{P(g_1,g_2)\cdot g_1\cdot g_2 - \mu_x\mu_y}{\sigma_x\sigma_y} \tag{6}$$

where μ_x, μ_y, σ_x and σ_y are the means and standard deviations of the marginal probabilities, found by summing the co-occurrence matrix rows (x) or columns (y).

The popular variant of co-occurrence uses grey-level differences, typically calculated for image samples of around 20^2 to 64^2 pixels (Parikh, 1977; Wu *et al.* 1985; Ebert, 1987; Tovinkere *et al.*, 1993), but also with 512^2 (Chen *et al.*, 1989). A grey-level difference $\Delta g(\theta,\delta)$ is the absolute difference between the grey-level at position (m,n) and the grey-level at position $(m+\phi(\theta,\delta), n+\phi(\theta,\delta))$, where the fixed spatial relationship ϕ is some radial function of the distance δ and direction θ from (m,n):

$$\Delta g(\theta,\delta) = |g(m,n) - g(m+\phi(\theta,\delta), n+\phi(\theta,\delta))| \tag{7}$$

Grey-level differences are calculated for every pair of pixels in the sample governed by $\phi(\theta,\delta)$, to produce a series of histograms $h_{\theta,\delta}(\Delta g)$. These histograms are then used to construct the grey-level difference statistics of interest, which commonly include:

$$\text{Mean}(\theta,\delta) = \frac{1}{G}\sum_{\Delta g=0}^{G-1} \frac{h_{\theta,\delta}(\Delta g)}{H}\cdot\Delta g \tag{8}$$

$$\text{Contrast}(\theta,\delta) = \sum_{\Delta g=0}^{G-1} \frac{h_{\theta,\delta}(\Delta g)}{H}\cdot\Delta g^2 \tag{9}$$

$$\text{ASM}(\theta,\delta) = \sum_{\Delta g=0}^{G-1} \left(\frac{h_{\theta,\delta}(\Delta g)}{H}\right)^2 \tag{10}$$

$$\text{Entropy}(\theta,\delta) = -\sum_{\Delta g=0}^{G-1} \frac{h_{\theta,\delta}(\Delta g)}{H}\cdot ln\left(\frac{h_{\theta,\delta}(\Delta g)}{H}\right) \tag{11}$$

where H is the total number of grey-level differences calculated for distance δ and angle θ. The mean and contrast measure the average and variability, respectively, of

grey-level differences in the sample, and produce large values for structured clouds such as cumulus with shadows. The angular second moment is a measure of the homogeneity in the sample, with a large value implying that although neighbouring pixels may have different grey-levels, the differences in these levels are similar over many pairs of pixels in the sample. This measure is high for cirrus or cumulonimbus anvils, for decks of stratus and bands of cloud oriented in the direction of θ. The entropy indicates the degree to which distinct scales of organisation are unrecognisable, and is a maximum when all grey-level differences Δg have an equal probability of occurring. When the texture is smooth or highly structured, the entropy measure is low (Wu *et al.*, 1985). In addition, some of the authors listed above have taken the mean and maximum values of these quantities over the various angles θ, as further cloud detection features.

Another class of textural measures that make use of spatial co-occurrence are known as grey-level run-length statistics (Haralick, 1986; Welch *et al.*, 1992). The method is based on the number of pixels found in a sample which share the same grey-level, and are collinear and connected. A co-occurrence density function $P(g,l)$ is used (similar to those above), but this time specifying the number of occurrences of runs of length l having grey-level g. If N_r is the number of runs in the sample, then run-length statistics are usually normalised with:

$$\Gamma = \sum_{g=0}^{G-1} \sum_{l=1}^{N_r} P(g,l) \tag{12}$$

Typical examples of run-length statistics include the short-run emphasis, long-run emphasis, grey-level nonuniformity, run-length nonuniformity and run percentage:

$$\text{SRE} = \frac{1}{\Gamma} \sum_{g=0}^{G-1} \sum_{l=1}^{N_r} \frac{P(g,l)}{l^2} \tag{13}$$

$$\text{LRE} = \frac{1}{\Gamma} \sum_{g=0}^{G-1} \sum_{l=1}^{N_r} P(g,l) \cdot l^2 \tag{14}$$

$$\text{GLN} = \frac{1}{\Gamma} \sum_{g=0}^{G-1} \left[\sum_{l=1}^{N_r} P(g,l) \right]^2 \tag{15}$$

$$\text{RLN} = \frac{1}{\Gamma} \sum_{l=1}^{N_r} \left[\sum_{g=0}^{G-1} P(g,l) \right]^2 \tag{16}$$

$$\text{RP} = \frac{1}{N_p} \sum_{g=0}^{G-1} \sum_{l=1}^{N_r} P(g,l) \cdot 100 \tag{17}$$

where the last statistic uses the total number of pixels N_p in the image as the normalisation constant. One of the main reasons for using such textural measures is to discriminate between large areas containing relatively uniform brightness values, such as sea ice and snow covered surfaces (Welch *et al.*, 1992).

The second types of textural feature described in Haralick (1986) are those obtained from gradient analysis, and generally refer to the amount of "edge" per unit

area, or *edge strength.* Wu *et al.* (1985) and Ebert (1987) both use the area averaged Roberts' gradient to assess edge strength, determined for each sample (consisting of a box of M by N pixels):

$$\mathrm{ARG} = \sum_{m=1}^{M-\delta} \sum_{n-1}^{N-\delta} \frac{|g(m,n) - g(m+\delta, n+\delta)| + |g(m+\delta, n) - g(m, n+\delta)|}{(M-\delta)(N-\delta)} \tag{18}$$

The distance δ is normally just the number of pixels over which the gradient is to be calculated, and is usually evaluated at $\theta = 0°$. The averaged Roberts' gradient calculated in this way is a maximum in regions of sharp brightness contrast.

Harris & Barrett (1978) used the technique of vector dispersion (a form of gradient analysis), which measures the dispersion in a 3-dimensional space of vector normals of a given sample of grey-levels. The technique is akin to measuring surface roughness, so that large values of the feature occur for smooth surfaces, whilst values near zero occur for uneven surfaces such as cellular cumuliform cloud.

Orthogonal transformations as a method of textural feature analysis refer to the way in which power spectra can be used to extract spatial frequency characteristics from images. Such features were considered early in studies of cloud classification from satellite imagery, and were first employed for meteorological purposes by Leese & Epstein (1963). More recent authors (Garand & Weinman, 1986; Ebert, 1987; Garand, 1988) all used the Fourier transform $g^*(u,v)$ of the image $g(m,n)$ over samples of M by N pixels:

$$\begin{aligned} g^*(u,v) &= \frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} g(m,n) \cdot \exp\left\{-2\pi i \left(\frac{mu}{M} + \frac{nv}{N}\right)\right\} \\ \text{with} \quad & u = 0, 1, \ldots, M-1 \\ & v = 0, 1, \ldots, N-1 \end{aligned} \tag{19}$$

Any significant maxima observed in the power spectrum intensity $|g^*(u,v)|^2$ were then used by Garand to identify specific orientations Φ and wavelengths λ characteristic of the sample, where:

$$\Phi = \tan^{-1}\left(\frac{u}{v}\right) \tag{20}$$

and

$$\lambda = \frac{Sl}{(u^2+v^2)^{1/2}} \tag{21}$$

The parameter $S = M = N$ is the linear dimension of the sample, assumed square, and l is the unit resolution of the pixels. In practice, Garand went on to use a feature called streakiness, based on a correlation coefficient estimated in the Fourier space (u,v), and the power spectrum intensity characterised by wavelengths in the range 20 to 40km. Ebert (1987) obtained the normalised average amplitude of the spectral components $g^*(u,v)$, for each wavenumber out to the Nyquist frequency of $S/\sqrt{2}$. She then calculated the logarithm of the mean and maximum of the normalised average amplitude, and the dominant wavenumber corresponding to the normalised average amplitude, as three textural features for cloud classification. In addition, Gu *et al.*

(1991) used three entropy measures in the spatial frequency domain by considering annuli at low, medium and high wavenumbers in the polar power spectrum as probability distributions. They also used a feature which measured the number of spectral components in the entire power spectrum greater than a critical censoring value.

Haralick (1986) also described a linear regression technique that has been used on the log of the power spectrum as a function of a general spatial frequency w, to estimate a fractal dimension D_1. For grey-level intensity surfaces of textured scenes which display fractal behaviour, the power spectrum satisfies:

$$g^*(w) \propto w^{-(2D_1+1)} \tag{22}$$

For smooth samples, Jones *et al.* (1989) pointed out that $D_1 \rightarrow 1$ (the dimension intuitively expected of a curve), whilst texturally rough surfaces resulted in values of $D_1 \rightarrow 0$ (note that their notation uses k instead of D_1). They also found that in Landsat images taken of cumuliform clouds over the sea, the fractal dimension D_1 varied according to the orientation over which their spatial frequencies were measured, such that small structures showed an elongation perpendicular to that for large structures (anisotropic scaling).

Transforms other than the Fourier transform have been compared for use in texture analysis, but researchers have reported no significant difference in the classification accuracy as a function of which transform was employed (Haralick, 1986). In general, Haralick also noted that some features based on Fourier power spectra have been shown to perform more poorly than features based on grey-level co-occurrence or difference statistics. This appears to be a result of aperture effects, which can be eliminated if measures such as regularity, directionality, linearity and coarseness are used (compare to the streakiness feature used by Garand (1988)). In this context, it is of interest to note that the seven spatial frequency measures analysed by Gu *et al.* (1991), and described above, were very poor in discriminating 13 cloud classes, except for stratus and thin cirrus over low cloud, for which one spatial frequency feature was the best!

The use of autocorrelation functions is another way of examining the spatial frequency characteristics of an image, and concerns the correlation of a copy of the image with itself. If the *tonal primitives* (areas of similar grey-level) on the image are relatively large, corresponding to coarse textures, then the autocorrelation drops off slowly with distance, whilst the opposite is true for small tonal primitives (Haralick, 1986). Given a sample of M by N pixels, and a translation (x, y), the autocorrelation function is formally defined as:

$$\rho(x,y) = \frac{1}{\Gamma} \cdot \frac{1}{(M-|x|)(N-|y|)} \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(m,n)g(m+x,n+y)\,dm\,dn \tag{23}$$

normalised by:

$$\Gamma = \frac{1}{MN} \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g^2(m,n)\,dm\,dn \tag{24}$$

assuming that the image has a mean value of 0, and that $|x| < M$ and $|y| < N$. The relationship between the autocorrelation function and the power spectral density is

well known: they are the Fourier transforms of one another. Gu *et al.* (1991) considered the use of a circular Moran autocorrelation function as a cloud classification feature, over four different scale lengths. Although they turned out to be poor discriminating features by themselves, they were found to be powerful in combination with other features, mainly because they respond to clouds of different size.

The remaining two statistical textural measures described in Haralick (1986) are mathematical morphology of tonal primitives and relative extrema density. Although mathematical morphology concerns the study of shape, it can be used for texture analysis, so that the shapes analysed are those of the tonal primitives. The usual approach requires the definition of a structuring element (a set of pixels constituting a specific shape such as a line, a disc or a square) and the generation of binary images, which result from the translation of the structuring element through the image, and the erosion of the image by the structuring element. The textural features can then be obtained by counting the number of pixels having the value 1. Textural features based on relative extrema densities are similar to those obtained using gradient analysis, except that instead of finding the edge strength, the number of extrema per unit area of the image is calculated. However, problems often arise with noise (smoothed images have been used), and the inability to differentiate between regions containing plateaus of extrema and many single-pixel extrema. It appears that neither of these feature types have previously been used in the field of cloud classification.

4.3 Spatial Features

All of the textural features discussed in the previous subsection are certainly concerned with spatial properties. However, it is important to distinguish between texture analysis, which regards non-random arrangements of tonal primitives in samples, and large scale spatial structure, perhaps occupying a significant fraction of an image, which is the subject of the present section. A good way of viewing the difference is to consider how a spatial feature might be calculated, compared to a textural feature. In textural analysis, samples of pixels are often taken as the basic data input, usually in the form of boxes of side 2^n pixels. Of course, all samples therefore have the same shape and size, so no spatial features of interest can be measured! Instead, spatial features can be thought of as measuring the shape and size of macro-pixel regions, defined by some common attribute. In this way, it can be seen that the use of spatial features is only of interest when an image has undergone some form of segmentation, perhaps having been previously classified with the help of textural analysis.

Segmentation is an important study in the field of pattern recognition in its own right, and is dealt with in more detail later in section 6. In fact, it will become apparent that there is a problem here for cloud system detection: should the image be classified according to boxed samples, segmented and reclassified using other features, or should segmentation take place first (perhaps using clustering techniques - §5.2), and then classified? These have become known as the "bottom-up" and "top-down" approaches to cloud classification (Peak & Tag, 1992), and will be discussed further in sections 6 and 7. To sum up, spatial features might be considered as features which operate at a much larger scale to that of textural features.

The size and shape of clouds have been an important source of information to

meteorologists for many years. In satellite imagery, growing cumulus may appear as single bright pixels at 1km resolution. When they develop to cumulonimbus in thunderstorms, they become much larger and are evident in visible and infrared images with sizes greater than 10km. A number of cumulonimbus clouds may become merged into a mesoscale convective boundary with a characteristic length of 100km or more. Mid-latitude low pressure areas or cyclonic storms, and fronts extending from them often have associated cloudy areas exceeding 1000km (Bunting & Hardy, 1984). The number of cloud systems characterised by shape are numerous, and include examples such as cloud streets, wave clouds (formed in the lee of mountains), jet stream cirrus (often having a long smooth curve close to the maximum winds at the poleward edge) and the high cloud associated with cyclogenesis (changing from an elongated leaf into a larger comma with cold frontal clouds in the tail of the comma).

Given such a rich diversity of this type of feature in clouds, it would seem logical that most past workers in cloud detection would have made use of them. However, there is very little evidence of this in the literature. One work that does stand out was that by Seddon (1983), who used a total of eight size and shape features in a pattern recognition analysis of atmospheric cloud features from Meteosat and Voyager images (the Voyager images taken from the encounter with Jupiter in March 1979). Her method consisted of an initial image segmentation, using clustering techniques, to produce a number of class related binary images, each containing value 1 for that class and 0 elsewhere. Two types of spatial feature were described, corresponding to whether they resulted in a scalar transform, or produced structural descriptions via space domain techniques.

Two scalar feature values were used and were based on the area of a bounded class region A and its perimeter length P. They were the thinness ratio t (see also Duda & Hart, 1973), and the coefficient of circular deficiency c, defined as:

$$t = \frac{4\pi A}{P^2} \tag{25}$$

$$c = \left| \frac{1}{t} - 1 \right| \tag{26}$$

The other shape features were all scalar measurements of the images, formed from space domain techniques. The first two were straightforward, being the area of each class A_{tot}, and the sum of the perimeter length in each class P_{tot}. For an image of size M by N pixels, the area of the class is just:

$$A_{tot} = \sum_{m=1}^{M} \sum_{n=1}^{N} g_b(m,n) \tag{27}$$

where $g_b(m,n)$ is the binary valued "grey"-level at position (m,n). The total perimeter length for each class is obtained from those pixels in the perimeter set P_s, that have a 0 valued neighbour:

$$P_{tot} = \sum_{m,n \in P_s} g_b(m,n) \tag{28}$$

The remaining four features were scalar measurements formed by medial axis transformation (Blum, 1967; Davis, 1986). In this process, a transform originates

at the boundary of objects and spreads away, either outwards or inwards, like a wavefront. For points which are the same distance away from two points on the boundary, discontinuities appear in the wavefront, and these form a so-called skeleton, which is the resultant image obtained from MAT.

Seddon (1983) only considered inner skeletons for calculating shape features, and used a series of eight erosion masks, partly because of the problem of correctly identifying a skeleton's minimal paths from digital data. Each mask, consisting of a central pixel set to 1, and 8 neighbours set to 0, 1 or either value, was used on every pixel in an image (except for the image border), such that if the nearest neighbours had the same values as the mask, the central pixel's value would be changed to 0. After all of the masks had been used, the image was examined for changes. If no pixels had changed, the skeleton had been found, otherwise all the masks were used again. Finally, the following four MAT features were calculated:

$$\mathrm{MAT}_1 = \sum_{m=1}^{M} \sum_{n=1}^{N} g_b^{\dagger}(m,n) \tag{29}$$

$$\mathrm{MAT}_2 = N_{\mathrm{pass}} \tag{30}$$

$$\mathrm{MAT}_3 = \frac{A}{(N_{\mathrm{pass}})^2} \tag{31}$$

$$\mathrm{MAT}_4 = \frac{P}{\mathrm{MAT}_1} \tag{32}$$

where $g_b^{\dagger}(m,n)$ was the binary value of each pixel after performing the MAT, and N_{pass} was the number of passes of the masks before there was no change in the image.

The size of an object is reflected in the number of pixels (MAT_1) and the number of passes required to obtain the skeleton in the transform (MAT_2). Feature MAT_3 gives an indication of how far the interior of an object is from the boundary (and is small if the object is "full"), whilst MAT_4 is small for complex shapes.

In the context of spatial features from clouds, the work of Burfeind *et al.* (1987) should be mentioned. They limited their analysis to the detection and classification of mature extratropical cyclones, using a sequence of pattern searches. Initially, they identified cores of extratropical cyclones with an infrared thresholding technique, by setting all pixels above the 60th percentile to 1, to create a binary image. A running mean filter was then used to reduce any "noise" present.

The next stage was to use a Sobel edge detector (similar to the Roberts' gradient discussed earlier), to produce an edge image. From this, a search was made for three boundaries: one at the northern edge of the cyclone, the second at a certain cutoff length on the eastern side, and a third corresponding to a horseshoe template, matched to the dry tongue of the cyclone in between the first two boundaries. The centre of the cyclone was then identified as a point 15% of the way from the template centre to the western edge of the dry tongue. In classifying the cyclone, two diagonal line were drawn through the centre, and the number of regions of cloudiness along each line estimated to determine the tightness of any spiralling present. These data were then used to classify the cyclone as one of type W, A, B, C, D_x or D_y (according to the scheme of Troup & Streten (1972)).

Again, the analysis in this case started with a form of segmentation (using the infrared thresholding technique), and went onto extracting features from shapes contained in the image (the centre of the cyclone and tightness of spiralling). The actual classification was then a matter of counting the number of spirals crossed (a pattern classification problem in 1 dimension!). Unfortunately though, this particular example is very specific to a limited number of objects in a satellite image, and an analysis using a much larger set of features would be required to detect a greater range of cloud system types.

One final spatial measure is well worth mentioning, which again seems to have skipped the attention of many authors. This is simply the geographical location of the sample, such as latitude (Bankert, 1994) or longitude. One could argue that these are really contextual measures, although they more closely relate to the family of spatial measures used by Peak & Tag (1992) which describe the latitudinal extent, longitudinal extent and northernmost latitude of synoptic scale cloud systems. In any case, contextual features are now discussed.

4.4 Contextual Features

In pattern recognition, the notion of context is incorporated explicitly in the form of dependencies between various elements, whether these elements are pixels, groups of pixels or larger scale attributes (Seddon, 1983). Producing a quantitative analysis of the context in images can be complex, but the relationships between elements can be used to solve problems such as error correction, ambiguity and gap filling.

Context can also be considered in terms of the various spatial relationships that exist in the field of syntactic pattern recognition (Fu, 1986). Unlike statistical pattern recognition where a pattern is represented by a feature vector, the syntactic approach represents a pattern as a composition of subpatterns or pattern primitives. The idea is analogous to the syntax of language, so that a "pattern grammar" might be considered to be a set of contextual rules concerning the structure of objects in an image. Since most cloud classification work has been directed with statistical techniques, this type of pattern recognition will not be considered in further detail. (It should be noted however, that Haralick & Fu (1983) demonstrated a method for discriminating cloud from various ground classes using a syntactic scheme based on cloud-shadow pair relationships).

For the cloud analyst, Bunting & Hardy (1984) stressed that contextual relationships such as wave clouds in the lee of mountains or cloud streets over oceans or large lakes were important. Another example quoted by them is that of cellular cloud patterns arising from convective mixing within a large air mass; this may be associated with the heating of the cold air behind a cold front, as it passes over warmer water. As with spatial features, it would appear that there might be a rich diversity of contextual features to use, but little research seems to have been undertaken in the field of cloud detection.

One notable exception was the work of Kittler & Pairman (1985), who used a contextual decision rule (see §5.1) based on a number of conditional joint probabilities, which themselves could be regarded as contextual features. As will be seen later, the *a posteriori* conditional probability $P(\omega_i|\underline{\xi})$ of obtaining a particular class ω_i given

a feature vector $\underline{\xi}$ plays an important rôle in statistical pattern recognition, and is related to the conditional probability $p(\underline{\xi}|\omega_i)$ according to Bayes's rule:

$$P(\omega_i|\underline{\xi}) = \frac{p(\underline{\xi}|\omega_i) \cdot P(\omega_i)}{p(\underline{\xi})} \tag{33}$$

where $p(\underline{\xi})$ is just the probability of obtaining feature vector $\underline{\xi}$. Kittler and Pairman used a contextual probability $P(\theta_0 = \omega_i|\underline{\xi}, \tilde{\underline{\eta}})$, with θ_0 being the class membership of the pixel under consideration, and $\tilde{\underline{\eta}}$ the class membership of the neighbourhood:

$$\tilde{\underline{\eta}} = [\theta_1, \ldots, \theta_l]^T \tag{34}$$

where l was the size of the context-conveying neighbourhood, and T denoted transposition. Clearly, the class membership of each pixel in the neighbourhood had to be obtained before the contextual probabilities could be computed, but as these were not known either, an initial estimate of the labels using a normal Bayesian classifier (again, see §5.1) was used instead. The resulting probabilities $P(\theta_0 = \omega_i|\underline{\xi}, \tilde{\underline{\eta}})$ were then used to obtain better estimates of $\tilde{\underline{\eta}}$ and so on, iteratively, until the process reached equilibrium. In doing so, they deduced a number of contextual joint probabilities $P(\tilde{\underline{\eta}}|\theta_0 = \omega_i)$, which in essence may be considered as contextual features.

Using an AVHRR image taken over the Atlantic Ocean in 1981, and a first-order neighbourhood only (4 vertical and horizontal neighbouring pixels), they obtained values of $P(\tilde{\underline{\eta}}|\theta_0 = \omega_i)$ for 8 training field classes. Most pixels of class membership $\theta_0 = \omega_i$ were surrounded by the same class (as expected), so that for class ω_1 (water), $P([1,1,1,1]\,|1)$ equalled 0.936 for example. Interestingly, for classes 4 through 8 (low, medium, thin cirrus, thick cirrus and deep convective cloud), these diagonal elements equalled 0.916, 0.838, 0.337, 0.639 and 0.440 respectively, indicating a certain lack of homegeneity for thin cirrus and deep convective cloud. Most other elements of this probability matrix were very small, with a mixture of thin and thick cirrus having the next largest value ($P([6,7,7,7]\,|7) = 0.041$). (Here, the order of the neighbour's classes is east pixel first, followed by the rest in an anticlockwise sense).

The decision rule turned out to be better than a standard Bayesian classifier, because it avoided noisy edges, thereby helping any later analysis of shape. An important point was that the use of context did not merely result in a smoothing effect: small real objects remained in the classified image if there was enough evidence of their existence. Another aspect of their work was that the technique did not have to rely on a first-order neighbourhood only. The neighbourhood could be expanded to include the whole image if necessary, with the use of regions rather than of individual pixels. This type of analysis on synoptic scales might well reveal some interesting contextual relationships.

There is one remaining feature of cloud detection, described at the beginning of this section in the list of features from Bunting & Hardy (1984). This is the presence of shadows, which can be useful for determining cloud height, and setting portions of the scene into context. However, it should now be apparent that details of shadowing can be found in terms of spectral, spatial and perhaps textural and contextual features. It might be better, therefore, to treat them as a type of cloud class initially, depending on the particular pattern recognition scheme adopted. In

fact, there is no reason why any class, once defined (and segmented), can be used as a higher order feature at a later stage of pattern recognition.

4.5 Feature Selection and Extraction

The need for feature selection and extraction arises from a number of concerns (Kittler, 1986). As a classifier increases in complexity with the number of feature space dimensions, it becomes important to base decisions on only the most essential, discriminatory information. Reducing dimensionality can also achieve a *decrease* in the error rate of recognition (the so-called peaking phenomenon: Duda & Hart §3.8, 1973; Kittler, 1986). Finally, if class overlap is too high in the feature space, then new sources of information will be required to improve class separability.

Strictly speaking, the methods of feature selection and feature extraction are quite different, although in many studies of cloud classification, both terms are used to refer to feature selection. Consider a number of features (usually obtained from amongst the training samples) which form the vector:

$$\underline{x} = [x_1, \ldots, x_D]^T \tag{35}$$

One way to reduce the vector dimensionality D is to identify and remove those features which do not contribute to class separability, leaving the optimal feature vector:

$$\underline{\xi} = [\xi_1, \ldots, \xi_d]^T \tag{36}$$

with smaller dimension d. This process is referred to as *feature selection* and is performed by calculating a criterion function J for a number of candidate feature vectors, each consisting of a subset d candidate features from the original D:

$$\underline{\xi}^c = [\xi_1^c, \ldots, \xi_d^c]^T \tag{37}$$

The optimal feature vector is then obtained by finding the maximum value of the criterion function over all possible candidate feature vectors:

$$J(\underline{\xi}) = \max_{\underline{\xi}^c} J(\underline{\xi}^c) \tag{38}$$

An alternative idea is to use all the features available (equation [35]), so that the useful information content is mapped via an optimal *feature extractor* A into the lower dimensional feature space. In this case, the optimisation is performed over all admissible mappings $\mathcal{A}$, found in the training set:

$$J(A) = \max_{\mathcal{A}} J(\mathcal{A}(\underline{x})) \tag{39}$$

Once A is determined, the optimal feature vector is easily found from:

$$\underline{\xi} = \mathcal{A}(\underline{x}) \tag{40}$$

Feature extraction has received very little attention as a means of reducing feature space dimensionality in cloud detection studies, and will not be considered further

here. A useful review of feature extraction techniques can however be found in the work by Kittler (1986).

In practice, and specifically in supervised classical procedures (§5.1), feature selection criterion functions are defined in terms of a number of model characteristics: the *a priori* class probabilities $P(\omega_i)$ for classes $\omega_i, i = 1, \ldots, m$, and the class conditional density functions $p(\underline{\xi}^c|\omega_i)$. In cloud classification schemes, the *a priori* probabilities are often assumed equal (Parikh, 1977; Wu *et al.*, 1985; Ebert, 1987), and are set to a value of 1. The class conditional density functions for the feature vector remain unknown until a training set of patterns is established, which define the class membership for each pattern. Again, in the field of cloud detection, a rather simplified density function has often been taken, in the form of multivariate normal (Gaussian) probability distributions (Kittler & Pairman, 1985; Wu *et al.*, 1985; Ebert, 1987; Garand, 1988):

$$p(\underline{\xi}^c|\omega_i) = \frac{1}{(2\pi)^{d/2}|\hat{\Sigma}_i|^{1/2}} \cdot \exp\left\{-\frac{1}{2}(\underline{\xi}^c - \underline{\hat{\mu}}_i)^T \hat{\Sigma}_i^{-1}(\underline{\xi}^c - \underline{\hat{\mu}}_i)\right\} \tag{41}$$

where $\underline{\hat{\mu}}_i$ is the maximum likelihood estimate of the vector of means $\underline{\mu}_i$ and $\hat{\Sigma}_i$ is the maximum likelihood estimate[2] of the covariance matrix Σ_i , for a total of N_i samples taken from class i:

$$\underline{\hat{\mu}}_i = \frac{1}{N_i}\sum_{n=1}^{N_i} \underline{\xi}_{in}^c \tag{42}$$

$$\hat{\Sigma}_i = \frac{1}{N_i - 1}\sum_{n=1}^{N_i} (\underline{\xi}_{in}^c - \underline{\hat{\mu}}_i)^T (\underline{\xi}_{in}^c - \underline{\hat{\mu}}_i) \tag{43}$$

The logic behind using maximum likelihood values is that there exist distribution parameters $\underline{\mu}_i$ and Σ_i whose values are fixed, but unknown. By using a training set, best estimate values for these parameters can be obtained (equations [42],[43]) which would predict the outcomes already observed (Duda & Hart, 1973). More recently however, the use of such normal distributions has been questioned (Benediktsson *et al.*, 1990; Lee *et al.*, 1990), particularly with regard to textural features.

One of the main objectives in designing a pattern recognition system is to classify unknown patterns with the lowest probability of misclassification (Kittler, 1986). It is also important to design a feature selection system in the same manner, using the minimum probability of error in classification e, which is given by:

$$e = \int \left[1 - \max_i \left\{P(\omega_i|\underline{\xi}^c)\right\}\right] p(\underline{\xi}^c)\, d\underline{\xi}^c \,. \tag{44}$$

For the 2-class case ($m = 2$), Bayes's rule (equation [33]) is used to obtain the relation

$$e = \frac{1}{2} \cdot \left\{1 - \int \left[p(\underline{\xi}^c|\omega_1)P(\omega_1) - p(\underline{\xi}^c|\omega_2)P(\omega_2)\right] d\underline{\xi}^c\right\} \tag{45}$$

[2]The notation used in equation [43] gives the unbiased form of the sample covariance matrix in order to avoid zero values of $|\hat{\Sigma}|$ (Duda & Hart, 1973; Thomas *et al.*, 1987).

When the integrand equals 1, or when the two functions $p(\underline{\xi}^c|\omega_i)$ do not overlap, the error is a minimum. The integral in equation [45] can be considered as a *probabilistic distance measure* between the two density functions, and criterion functions of this form are probably the most popular for feature selection using remote sensing data. Kittler (1986) listed a total of ten such distance measures for the 2-class case, three of the most commonly used being:

$$\text{Bhattacharyya} \quad J_B = -\ln \int \left[p(\underline{\xi}^c|\omega_1)p(\underline{\xi}^c|\omega_2)\right]^{1/2} d\underline{\xi}^c \tag{46}$$

$$\text{Divergence} \quad J_D = \int \left[p(\underline{\xi}^c|\omega_1) - p(\underline{\xi}^c|\omega_2)\right] \cdot \ln \left[\frac{p(\underline{\xi}^c|\omega_1)}{p(\underline{\xi}^c|\omega_2)}\right] d\underline{\xi}^c \tag{47}$$

$$\text{Matusita} \quad J_M = \left\{\int \left[\sqrt{p(\underline{\xi}^c|\omega_1)} - \sqrt{p(\underline{\xi}^c|\omega_2)}\right]^2 d\underline{\xi}^c\right\}^{1/2} \tag{48}$$

For multivariate normal distributions, the 2-class probabilistic distance measures can be evaluated in terms of the four maximum likelihood parameters $\hat{\underline{\mu}}_1$, $\hat{\underline{\mu}}_2$, $\hat{\Sigma}_1$ and $\hat{\Sigma}_2$, and therefore have the advantage that they can be computed recursively:

$$J_B = \frac{1}{4} \cdot (\hat{\underline{\mu}}_1 - \hat{\underline{\mu}}_2)^T \left\{\hat{\Sigma}_1 + \hat{\Sigma}_2\right\}^{-1} (\hat{\underline{\mu}}_1 - \hat{\underline{\mu}}_2) + \frac{1}{2} \cdot \ln \left\{\frac{\left|\frac{1}{2}(\hat{\Sigma}_1 + \hat{\Sigma}_2)\right|}{\sqrt{|\hat{\Sigma}_1||\hat{\Sigma}_2|}}\right\} \tag{49}$$

$$J_D = \frac{1}{2} \cdot (\hat{\underline{\mu}}_1 - \hat{\underline{\mu}}_2)^T \left\{\hat{\Sigma}_1^{-1} + \hat{\Sigma}_2^{-1}\right\} (\hat{\underline{\mu}}_1 - \hat{\underline{\mu}}_2) + \frac{1}{2} \cdot \text{tr}\left\{(\hat{\Sigma}_1 - \hat{\Sigma}_2) \cdot (\hat{\Sigma}_2^{-1} - \hat{\Sigma}_1^{-1})\right\} \tag{50}$$

$$= (\hat{\underline{\mu}}_1 - \hat{\underline{\mu}}_2)^T \hat{\Sigma}^{-1} (\hat{\underline{\mu}}_1 - \hat{\underline{\mu}}_2) \qquad \text{if } \hat{\Sigma}_1 = \hat{\Sigma}_2 = \hat{\Sigma} \tag{51}$$

$$J_M = \sqrt{2\left[1 - \exp\left\{-J_B(\hat{\underline{\mu}}_1, \hat{\underline{\mu}}_2, \hat{\Sigma}_1, \hat{\Sigma}_2)\right\}\right]} \tag{52}$$

(The criterion function obtained in equation [51] for the special case $\hat{\Sigma}_1 = \hat{\Sigma}_2$ is the well-known Mahalanobis distance). Figure 2 shows how another distance measure (Euclidean), could be used to estimate the separability of two normally distributed classes for two different features. Using the same parameters as above, the Euclidean distance measure (which is independent of covariance, and hence rather limited for real multi-dimensional distributions) is defined by:

$$J_E = (\hat{\underline{\mu}}_1 - \hat{\underline{\mu}}_2)^T (\hat{\underline{\mu}}_1 - \hat{\underline{\mu}}_2) \tag{53}$$

Note that the distance measures given in equations [49] to [52] are similar, but are additionally weighted by the covariance matrices, in a number of ways.

According to Lillesand & Kiefer (1987), many mathematical expressions for the divergence criterion J_D exist, although Thomas *et al.* (1987) and Gu *et al.* (1991) based their transformed divergence $J_{D_{ave}}$ (extended from 2 to m classes) on the definition given above in equation [50]. In other cloud classification work, the feature selection criterion most often used in the past was the 2-class Fisher distance (Parikh,

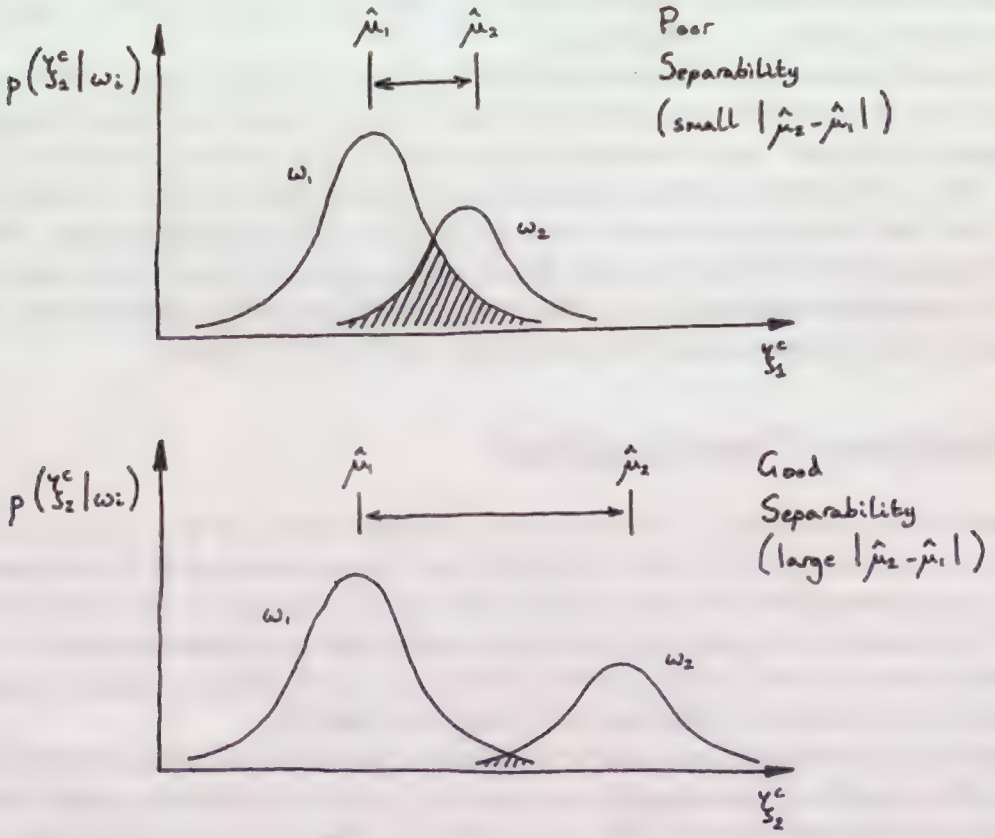

Figure 2: *An example of how a probabilistic distance measure can be used to evaluate the separability of two classes ω_1 and ω_2 for two candidate features ξ_1^c (top) and ξ_2^c. The class conditional probability distributions are assumed normal, and the measure is simply the difference of means $|\hat{\mu}_2 - \hat{\mu}_1|$, which is the square root of the Euclidean distance measure in 1 dimension (equation* [53]*). The measure indicates that ξ_2^c will yield fewer misclassifications (the shaded areas) than ξ_1^c.*

1977; Gu *et al.*, 1991), also sometimes called the divergence (Wu *et al.*, 1985; Ebert, 1987; Garand, 1988) and evaluated for a single feature:

$$J_F = \frac{|\hat{\mu}_1 - \hat{\mu}_2|}{\hat{\sigma}_1 + \hat{\sigma}_2} \tag{54}$$

where $\hat{\sigma}_1$,$\hat{\sigma}_2$ are the maximum likelihood standard deviations of the feature for classes 1 and 2 respectively. In other work, Duda & Hart (1973), Fukunaga (1986) and Gu & Duncan (1988) used what they respectively called the Fisher linear discriminant function, the Fisher criterion and the F-ratio, (sometimes referred to as the ratio of between-class scatter to within-class scatter):

$$J'_F = \frac{(\hat{\mu}_1 - \hat{\mu}_2)^2}{\hat{\sigma}_1^2 + \hat{\sigma}_2^2} \tag{55}$$

Most of these authors then ranked their feature selection criteria in order of their ability to separate, as indicated in equation [38]. Ebert (1987) and Garand (1988)

also used artificial samples obtained with a random number generator (having the same means and variances as the real data) to yield the theoretical limit of separability of the defined classes as a comparison.

For non-parameterised features, a further stage of analysis might be to consider searching for feature sets with methods such as the branch and bound algorithm, the sequential forward and backward algorithms or the max-min method (Kittler, 1986). This is because the search becomes a combinatorial problem of trying to find one optimal set of features from a total of $D!/(D-d)!d!$ candidate sets. However, using training sets to estimate model distribution parameters such as μ_i and σ_i cuts down the required computation significantly, and is one of the reasons why the use of normal distributions and pairwise selection has been so popular.

5 Pattern Classification

The detail for constructing a d-dimensional feature space for the purpose of cloud pattern recognition has now been presented, and many examples of the features used in real cloud recognition schemes will be reviewed in section 7. However, the next part of the process is to partition the feature space into the classes required. It is the user's definition of "class" that controls the decision of whether to employ supervised or unsupervised learning techniques to obtain the classifier.

Supervised techniques can only be used when the classes have been labelled before training the classifier. For example, an image to be used for training might have had a certain set of regions classified by an expert into different classes of interest. These regions generally consist of "pure" samples of the class under consideration, carefully located to avoid pixels along edges adjacent to other classes, a technique described by some as both an art and a science (Lillesand & Kiefer, 1987).

On the other hand, unsupervised learning considers the classes to be unlabelled before training. During training, it is found that "natural" groupings form in the feature space, corresponding to different classes that the data suggest. It is then the job of the expert to identify these groupings and label the resultant classes. The classification during the training stage is usually performed with a cluster analysis technique, although later classification may be done with supervised techniques such as the minimum distance method (Farki *et al.*, 1992). In fact, such combinations of supervised and unsupervised analyses have been suggested as the preferable method (Swain, 1986).

Later, it will become apparent that artificial intelligence approaches, such as neural networks, appear to hold great promise for the classification of satellite imagery, because of their skill in learning the characteristics of data and their ability to define nonlinear discriminant functions. For now, however, the "classical" methods of pattern classification must be addressed.

5.1 Supervised Classical Techniques

In the supervised approach, every feature vector in the training data has a label belonging to a specific class, and clusters of these vectors will tend to occur in the

feature space. Somehow, the clusters need to be defined explicitly or the feature space needs to be partitioned, so that new feature vectors can be assigned a class.

The way in which the feature space is partitioned depends on whether each cluster can be considered to form a parametric distribution, or whether it should remain a nonparametric set of individual data points. If the total number of feature vectors used in the training stage is N_T, each with d dimensions, then a total of $(N_T)^d$ storage locations will be required for a nonparametric analysis. This is one of the advantages of parametrising the clusters: the normal distribution of equation [41] requires only $m2^d$ parameters for m classes. In discussing nonparametric techniques, (k) nearest neighbour, parallelepiped and discriminant function classifiers will be covered. Parametric techniques such as minimum-distance-to-means, Gaussian maximum likelihood (or Bayesian) classifiers will be considered toward the end of this section.

The most basic form of supervised classifier relies on **Nearest Neighbour Techniques**. Any unclassified feature vector is simply assigned to the class belonging to the nearest neighbour in the feature space, usually by finding the minimum Euclidean distance:

$$D_{NN} = \min_{n} \sqrt{\sum_{j=1}^{d} (\xi_j - \xi_{nj})^2} \qquad \text{for} \quad n = 1, \ldots, N_T \tag{56}$$

where ξ_j is the jth element of the d-dimensional feature vector to be classified[3] and ξ_{nj} is the jth element of the nth training feature vector.

Such techniques do however have major drawbacks. For example, a "rogue" training pattern will result in misclassification for any new feature vector for which it is the nearest neighbour (see figure 3a). One solution to this problem is to find the nearest k neighbours, and classify the new feature vector according to the most common class found amongst those k neighbours. Interestingly, it can be shown (Duda & Hart, 1973) that with an unlimited number of training samples, the error rate of classification for the nearest neighbour rule is never worse than twice the error rate for the optimal (Bayesian) procedure (see later).

Another simple nonparametric technique which partitions the feature space is the **Parallelepiped Classifier**. Here, some sensitivity to the variance of features in a class cluster is achieved by considering the range of feature values over which the training data are found (figure 3b). This type of classifier is often used to produce a "quick-look" scene analysis, especially if a good interactive colour display system is available for training (Swain, 1986). Although it has the advantage of being easy to train, the results are very dependent on analyst bias, and it is quite likely that some new feature vectors will be classified into more than one parallelepiped as a result of class overlap (again, see figure 3b). This last problem may be overcome by assigning classes to smaller connected parallelepipeds having stepped borders (Lillesand & Kiefer, 1987). A further problem however, is that some new feature vectors may well lie outside the parallelepipeds, and therefore be impossible to classify.

Instead of trying to box up the feature space into class cluster regions, surely

[3]Note that the superscript c in $\underline{\xi}^c$ has now been dropped, as the optimal set of candidate feature vectors are assumed to have been found.

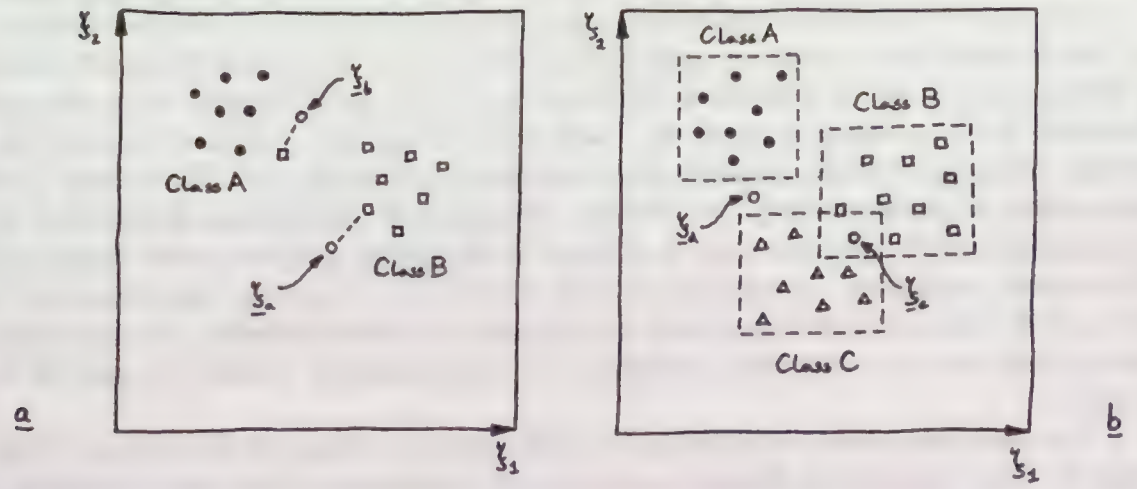

Figure 3: *Examples of nonparametric supervised classifiers used in a 2-dimensional feature space.* **a:** *Nearest neighbour - $\underline{\xi}_b$ was misclassified due to the rogue from class B near cluster A.* **b:** *Parallelepiped - there are problems in trying to classify $\underline{\xi}_c$, as it occurs in a region of 2 overlapping classes, and $\underline{\xi}_d$ cannot be classified.*

it would be better to draw a dividing boundary between each cluster, which would allow more room for new feature vectors from a given class and would classify all possible positions in the feature space. This can be achieved with the use of decision boundaries or equivalently, **Discriminant Functions**. Such functions are able to map any feature vector into the feature space in terms of the boundaries, and result in an immediate classification based on the value of the function.

One of the major problems facing the use of discriminant functions however, is the total number of solutions that might exist. If there are d features (each having a discrete range from 1 to r) and m classes, then there are approximately $m^{r^d}/m!$ possible discriminant functions (Duda & Hart, 1973): for only 2 features, each displaying 20 values, a 2-class problem contains over 10^{120} potential partitions! If however, a linear classifier is used to partition the feature space, only a small number of these mappings can be used, but the classes must be *linearly separable* (the decision surface in d dimensions must be a hyperplane).

Consider a linear decision boundary separating two classes in a 2-dimensional feature space (figure 4). It can be described by a weight vector $\underline{w}$ and a threshold weight w_0 (or bias), so that the discriminant function $f(\underline{\xi})$ is given by:

$$\begin{aligned} f(\underline{\xi}) &= \underline{w}^T \cdot \underline{\xi} + w_0 \\ &= \sum_{j=1}^{d} (w_j \xi_j) + w_0 \end{aligned} \tag{57}$$

A two category linear classifier might therefore use the decision rule: decide class ω_1 if $f(\underline{\xi}) > 0$ and class ω_2 if $f(\underline{\xi}) < 0$. The decision boundary $f(\underline{\xi}) = 0$ defines a straight line in the case $d = 2$, with gradient $-w_1/w_2$ and intercept $-w_0/w_2$ (by comparing to $y = mx + c$). A more useful weight vector to use is the $(d+1)$-dimensional vector

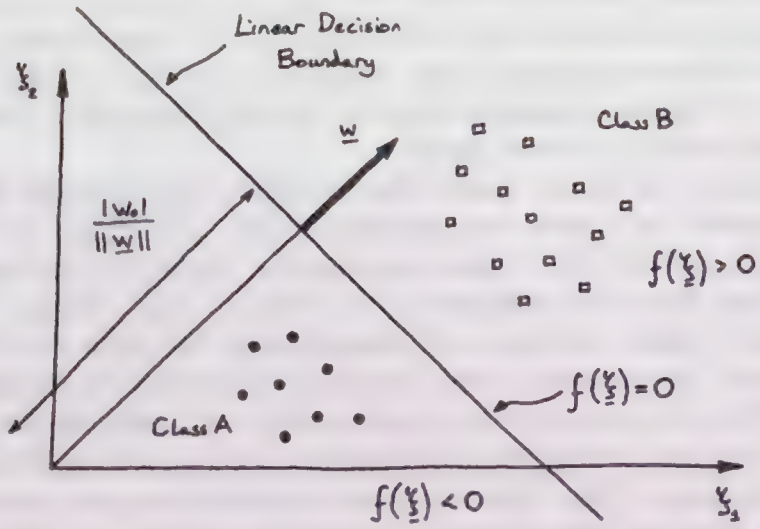

Figure 4: *A linear decision boundary between 2 classes in a 2-dimensional feature space ξ_1, ξ_2. The weight vector $\underline{w}$ can be used to define the geometry of the decision boundary, which is expressed in the form of a discriminant function $f(\underline{\xi})$.*

$\underline{w}'$, with elements $w_0, w_1, \ldots, w_d$ and feature vector element $\xi_1' = 1$, such that:

$$f(\underline{\xi}) = \sum_{j=1}^{d+1} w_j' \xi_j' \tag{58}$$

To obtain the required classifier, the modified weight vector $\underline{w}'$ must be determined from the training set. Unfortunately, this is not trivial, even in a two category, linearly separable situation. Typical methods are to define a criterion function $J(\underline{w}')$ which is minimised when $\underline{w}'$ is a solution vector, and to use an iterative gradient descent procedure to modify the weights for each new feature vector from the training set (Duda & Hart, 1973); a pictorial representation of this technique will be given later in §5.3. The simplest form of gradient descent procedure is:

$$\underline{w}'(t+1) = \underline{w}'(t) - \rho \nabla J(\underline{w}'(t)) \tag{59}$$

where t represents the iteration number in the procedure and ρ is a positive scale factor that sets the step size. An interesting criterion function which will be considered again later in §5.3 is the "Perceptron Criterion Function", developed by Rosenblatt (1962) as a reinforcement learning scheme in neural networks. This is based on the set $\underline{S}$ of samples *misclassified* by $\underline{w}'$:

$$J_P(\underline{w}') = \sum_{\underline{\xi}' \in \underline{S}} \sum_{j=1}^{d+1} (-w_j' \xi_j') \tag{60}$$

Numerous other gradient descent procedures, each using a particular set of criterion functions, have been listed by Duda & Hart (1973) and include the relaxation, minimum squared error, Ho-Kashyap and linear programming techniques. Unfortunately,

they are rather limited: relaxation procedures converge to a solution vector for the linearly separable case, but do not show convergence otherwise. The Ho-Kashyap and linear programming methods go a little further and show evidence of nonlinear separability, and minimum squared error procedures provide a weight vector in all cases, but with no absolute guarantee that the resulting "solution" vector separates the classes in the linearly separable case!

In multicategory situations, more than one linear discriminant function is required. It is possible to reduce the problem to $(m-1)$ 2-class problems, where the ith problem is solved with a linear discriminant function that separates points assigned to ω_i from those not assigned to ω_i (Duda & Hart, 1973). The resulting classifier is called a linear machine, and interestingly, this idea has recently resurfaced in the form of "don't care" neural networks (see section 7). Another extension to the general idea is to introduce any nonlinearity that exists in the separation of classes into the decision surface itself, by applying a nonlinear transformation to the data (Beale & Jackson, 1992). Following the transformation, the patterns are then capable of being classified using linear discriminant functions and the technique is referred to as a Φ machine.

So far, all the techniques considered have been nonparametric. It would seem reasonable to suppose that some statistical measure for each class cluster might be useful as an aid to partitioning the feature space. For example, an obvious step to improving the nearest neighbour classifier described earlier is to calculate the distance between any new feature vector and the *mean* of the class clusters. This parametric approach is called **Minimum-Distance-to-Means Classification**, as it uses the mean vector $\hat{\underline{\mu}}_i$ for each class $i = 1,\ldots,m$ obtained from the training set, (defined in equation [42]). In this case, an unclassified feature vector is assigned the class of the nearest mean vector (see figure 5a), and for a Euclidean distance classifier, this is simply:

$$D_\mu = \min_i \sqrt{\sum_{j=1}^{d}(\xi_j - \hat{\mu}_{ij})^2} \qquad \text{for} \quad i = 1,\ldots,m \tag{61}$$

However, the major problem with the Euclidean distance measure is that it is insensitive to differing degrees of variance within each class cluster. For example, consider a well defined (low variance) class cluster which happens to be nearby to a high variance cluster (figure 5a). A new feature vector lying between the two means, but slightly nearer to the low variance cluster mean will be assigned to that cluster, even though it may easily belong to the higher variance cluster. One way of avoiding this problem is to use the Mahalanobis distance presented in equation [51]. In this case, the distance is modified by dividing it with the variance of the class in the appropriate d-dimensional (point to mean) direction (Thomas *et al.*, 1987).

The **Gaussian Maximum Likelihood Classifier** is by far the most commonly used classifier in remote sensing (Swain, 1986). Each class cluster in the training set can be described by some probability distribution, the parameters of which can be evaluated during the training phase. The method of classifying a new feature vector is to calculate values from the probability distribution for every class, and to assign the feature vector to the class with the largest value (figure 5b).

Before describing the technique in detail, it is worth discussing some of the ter-

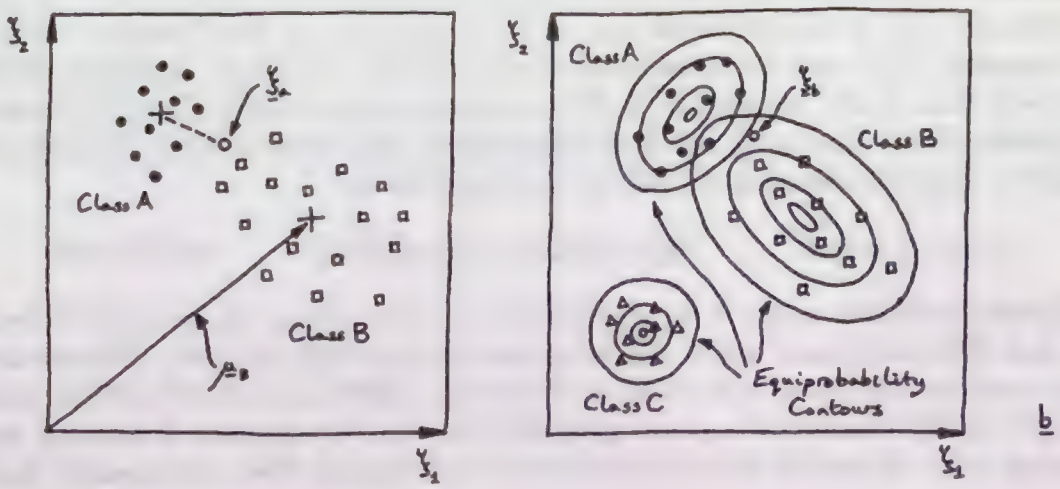

Figure 5: *Examples of parametric supervised classification techniques in a 2-dimensional feature space.* **a:** *Minimum-distance-to-means (the mean vector for each class cluster is shown by a cross) - $\underline{\xi}_a$ is misclassified in this example, because a Euclidean distance has been chosen which takes no account of variance.* **b:** *Maximum likelihood estimates of normal distribution model parameters - in this case, the optimal Bayesian classifier can be used, providing the model assumptions are correct.*

minology. Earlier, in §4.5, the term *maximum likelihood* was used to refer to the best estimate values for the mean and covariance parameters of a normal (Gaussian) distribution, in the sense that they maximise the class conditional probability $p(\underline{\xi}|\omega_i)$ of obtaining the samples already observed (see equations [41] to [43]). Some authors prefer to use the term likelihood to refer to *a posteriori* probabilities of the form $P(\omega_i|\underline{\xi})$ (Thomas *et al.*, 1987; Beale & Jackson, 1992), which can cause some confusion. Here, the meaning of maximum likelihood estimation and Bayesian estimation is the same as that given by Duda & Hart (1973): "Maximum likelihood methods view the parameters as quantities whose values are fixed but unknown. The best estimate is defined to be the one that maximises the probability of obtaining the samples actually observed. Bayesian methods view the parameters as random variables having some known *a priori* distribution. Observation of the samples converts this to an *a posteriori* density, thereby revising our opinion about the true values of the parameters".

In using supervised statistical methods in pattern recognition, it is the *a posteriori* probability $P(\omega_i|\underline{\xi})$ that is of interest: given a feature vector $\underline{\xi}$, what is the probability of it belonging to class ω_i? Clearly, the statistical approach would suggest that $\underline{\xi}$ should belong to the particular class ω_i for which $P(\omega_i|\underline{\xi})$ is the greatest. This is known as the Bayesian classifier:

$$\text{Assign } \underline{\xi} \text{ to } \omega_i \text{ if} \qquad P(\omega_i|\underline{\xi}) > P(\omega_j|\underline{\xi}) \qquad \text{for all } j \neq i \tag{62}$$

In terms of the smallest rate of misclassification (discussed in §4.5), this turns out to be the best estimate possible. In practice however, such *a posteriori* probabilities are usually not known, and an estimate must be obtained by some other means (unless a recursive Bayesian learning method is used (Duda & Hart, 1973), which is rather

uncommon). This is where the use of maximum likelihood values enters. If a model distribution for a feature vector can be assumed (such as the Gaussian distribution of equation [41]), and its parameters can be estimated (with maximum likelihood values), then it can be related to the *a posteriori* probability through Bayes's rule (equation [33]). Since $p(\underline{\xi})$ are class independent, they behave simply as scale factors, and the Bayesian classifier of [62] can be rewritten as:

$$\text{Assign } \underline{\xi} \text{ to } \omega_i \text{ if} \qquad p(\underline{\xi}|\omega_i)P(\omega_i) > p(\underline{\xi}|\omega_j)P(\omega_j) \qquad \text{for all } j \neq i \tag{63}$$

A special condition arises if the *a priori* probabilities $P(\omega_i)$ are assumed to be equal, so that different classes within the scene are equally likely to occur. This assumption was used for the cloud classifiers of Wu *et al.* (1985) and Ebert (1987), although Parikh (1977) calculated *a priori* probabilities from the number of training samples in each class. If equal *a priori* probabilities are assumed, then the classifier becomes:

$$\text{Assign } \underline{\xi} \text{ to } \omega_i \text{ if} \qquad p(\underline{\xi}|\omega_i) > p(\underline{\xi}|\omega_j) \qquad \text{for all } j \neq i \tag{64}$$

The Gaussian assumption for the form of the class conditional densities has nearly always been used, because it is has been considered "convenient and adequate" (Kittler & Pairman, 1985) and has been well researched. If some classes are found that violate the normal distribution assumption[4], the problem is often dealt with by subdividing those classes into subclasses, each represented by its own normal distribution (Swain, 1986). If the Gaussian assumption for the class conditional densities is considered valid, then the logarithm of $p(\underline{\xi}|\omega_i)$ is sometimes used in the classification algorithm, as it is computationally more stable (Fukunaga, 1986; Ebert, 1987; Thomas *et al.*, 1987):

$$\ln\left[\, p(\underline{\xi}|\omega_i) \,\right] = -\frac{d}{2}\ln 2\pi - \frac{1}{2}\ln|\hat{\Sigma}_i| - \frac{1}{2}(\underline{\xi} - \underline{\hat{\mu}}_i)^T \hat{\Sigma}_i^{-1}(\underline{\xi} - \underline{\hat{\mu}}_i) \tag{65}$$

It should be noted however, that a few authors have preferred to use normalised histograms of the training data set in order to estimate the density functions $p(\underline{\xi}|\omega_i)$ (Benediktsson *et al*, 1990). These are often used as substitutes where approximations to a normal distribution are very poor, and calculations using this approach are of course more time consuming due to the nonparametric nature of the problem. A good discussion of the use of nonparametric techniques to obtain the density distributions (such as Parzen windows and k nearest neighbour estimates) was given by Duda & Hart (1973).

Another possible disadvantage of Gaussian maximum likelihood methods is that the classification process is much slower than any of the previous techniques and the computation time varies approximately as the square of the number of features used (Swain, 1986; Lillesand & Kiefer, 1987). In addition, because second-order statistics are required to fully parameterise the distributions, more training data are needed than for other parametric methods such as the minimum-distance-to-means classifier.

Before leaving the subject of supervised classification, it is worth describing some further "techniques" that may be used in training, and which can help reduce the

[4]There have been objections to such general use of the normal assumption more recently, especially with regard to textural features (see §4.5) and its redundancy in artificial neural networks.

number of eventual misclassifications. The first aspect concerns the size of the training set required to perform a sufficiently accurate classification. Theoretically, the lower limit is $d+1$, where d is the feature space dimension (Lillesand & Kiefer, 1987). In practice however, it has been found that a minimum of $10d$ to $100d$ samples are needed to obtain a suitable statistical significance. Tovinkere *et al.* (1993) prescribed an alternative sample minimum based on the number of classes m. Their rule-of-thumb was that $10m$ to $20m$ samples should be used for each class, and in many past cloud classifiers, this has often been the case (see section 7).

If the probability of obtaining a particular feature vector is low, then classification may not result in a valid representation of the class to which it would be assigned, and it might be better to ignore the result altogether. This can be achieved by thresholding (Thomas *et al.*, 1987), so that a feature vector which shows a maximum probability of occurring less than 1% of the time (for example) would be filtered out of the classification process. A 1% threshold usually corresponds to a feature being removed some 3σ from the mean, in 1 dimension for a homogeneous class. For Gaussian statistics, this corresponds to a probability of only 0.2% that the feature is correctly located outside the 1% boundary.

Another technique aimed at reducing misclassification is the use of a confusion matrix (Lillesand & Kiefer, 1987). The idea is to classify the same feature vectors used in the training, *after* the training has been performed. A confusion matrix is simply a table containing the number of known feature vectors belonging to each class listed against the number obtained from the classifier for each class. Ideally, all off-diagonal elements of the confusion matrix are zero, resulting in an overall accuracy of 100%. In reality, nonzero off-diagonal elements are found, and analysis of those elements can provide insight into aspects such as poorly defined classes, classes with low statistical significance and features which have poor separability. Examples of the use of confusion matrices may be found in work by Ebert (1987), Garand (1988), Lee *et al.* (1990) and Tovinkere *et al.* (1993).

5.2 Unsupervised Classical Techniques

It may not always be possible to label a set of training data, and in any case, the labelling by experts may yield results which show great variation and subjectivity. Specifically *not* labelling the training data may prove to be more advantageous, especially if there is a tendency for the features to form "natural" clusters in the d-dimensional feature space.

Unsupervised learning concerns the training of a classifier which is based on the analysis of such unlabelled data, usually through the methods of cluster analysis. Cluster analysis can be defined as one of the tools of exploratory data analysis "that attempts to assess the interactions among patterns by organising the patterns into groups or clusters, such that patterns within a cluster are more similar to each other than are patterns belonging to different clusters" (Jain, 1986).

Once valid clusters have been determined from the training data, the job of the analyst is to identify which clusters belong to which classes in the scene, so that any new feature can be correctly assigned to one of those classes. This step has often been performed using supervised techniques such as the minimum-distance-to-means

method, described in §5.1 (Seddon & Hunt, 1985; Farki *et al.*, 1992), resulting in a hybrid classification scheme. The use of training data however, is not always implied as a means to provide a new classifier. Since it is an exploratory method, the aim may be to gain insight into the nature or structure of the data and to discover new classes or subclasses (Desbois *et al.*, 1982), to search for regions within target areas that may be used for cloud wind vector studies (Wade *et al.*, 1992), or even to assess different cluster algorithms themselves (Pairman & Kittler, 1986).

Before looking at individual clustering techniques, it is worth considering what goes to make an ideal cluster, since there has been concern about questions such as "Are the data random?" and "How many clusters are there?" (Jain, 1986). These questions also come under the domain of cluster validity, which will be considered in more detail toward the end of this section.

Jain (1986) considered three definitions of ideal clusters. With reference to a d-dimensional feature space, clusters should contain features which are similar, so that features belonging to different clusters are less similar. They should also consist of an aggregation of features such that the distance between any two features in the cluster is less than the distance between a feature inside and a feature outside the cluster. Finally, ideal clusters should be described as connected regions of feature space containing a relatively high density of features, separated from each other by regions containing a relatively low density of features. The clustering method chosen should in practice depend on such notions of ideal clusters, although it is important to remember that if the data do not satisfy the assumptions made, then a clustering technique may *impose* structure on the data rather than uncover the true structure.

Most clustering algorithms in the literature can be classified into one of two types: hierarchical and partitional. Hierarchical algorithms will only be discussed briefly, since most unsupervised schemes used to examine cloud in satellite imagery have relied on partitional methods, which can be considered to be unsupervised analogues of the popular parametric techniques used in supervised classification.

As the name suggests, **Hierarchical Techniques** consist of a sequence of partitions between clusters which can range from the total of N_T clusters, one for every training sample, to a single cluster for the whole set of samples. Classical examples of hierarchical clustering appear in biological taxonomy, where individuals are grouped into species, species into genera, genera into families and so on (Duda & Hart, 1973), resulting in a *dendrogram* of how the samples are grouped (figure 6). If the hierarchical clustering is performed from the N_T clusters initially and grows a sequence of clusters until a single cluster is reached, the process is termed agglomerative. The opposite approach is naturally termed divisive.

The way in which an agglomerative procedure works is roughly as follows. First, all N_T training features are assigned to N_T unique clusters. A measure of the dissimilarity between each cluster is calculated and added to a dissimilarity matrix, so that when all the calculations are complete, the smallest entry is found. The corresponding clusters in the matrix are merged and the dissimilarities between the remaining $N_T - 1$ clusters are calculated. This last process is repeated for $N_T - 2$, $N_T - 3$ clusters and so on, until all the training features fall into one cluster.

Most of the commonly used updating procedures can be expressed in a single formula which gives the dissimilarity D_H between a cluster k and a cluster $i + j$

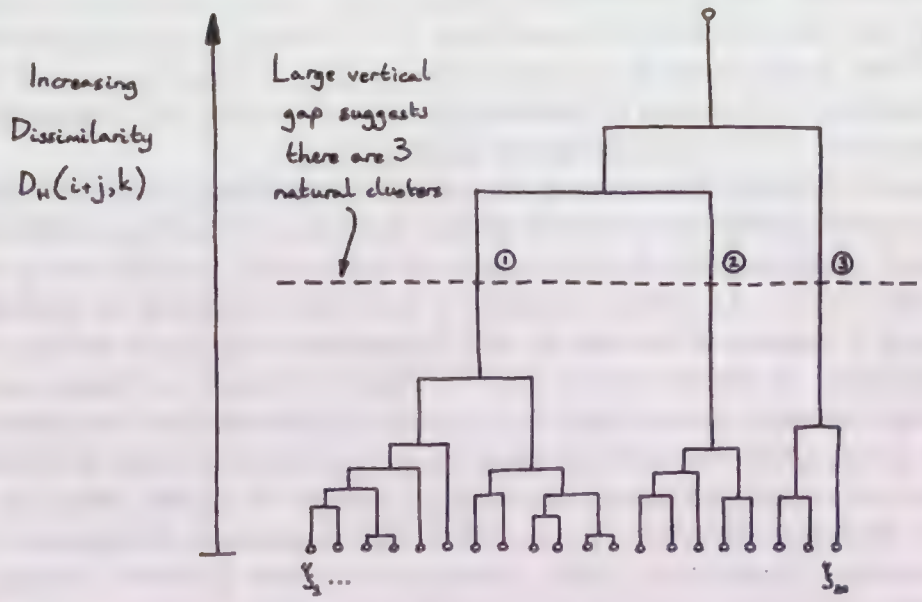

Figure 6: *A dendrogram obtained using an agglomerative hierarchical clustering algorithm. The training features are represented by individual clusters at the bottom, and the vertical axis gives the level of dissimilarity between clusters. A suitable partition is found where the largest vertical gap occurs, suggesting that there are 3 clusters here.*

formed by the merger of two clusters i and j (Jain, 1986):

$$D_H(i+j,k) = a_i D(i,k) + a_j D(j,k) + bD(i,j) + c|D(i,k) - D(j,k)| \qquad (66)$$

where $D(i,j)$ is simply the distance between two clusters i and j, and a_i, a_j, b and c are parameters whose values depend on the exact clustering algorithm chosen.

When $a_i = a_j = 0.5$, $b = 0$ and $c = -0.5$, the process determines the smallest of the minimum distances between each cluster (the nearest neighbour) and is called the single link algorithm. One of the behavioural characteristics of the single link algorithm is that features further removed from each other are put into the same cluster because they share a neighbour. This is often called the "chaining effect" (Duda & Hart, 1973) and can be a problem due to its sensitivity to noise. However, if there is a tendency for the data to form chains, the algorithm can be advantageous.

Another well known case is when $a_i = a_j = 0.5$, $b = 0$ and $c = 0.5$. This is the complete link or diameter algorithm, and the smallest of the maximum distances between each cluster is used. The effect discourages the growth of long clusters and encourages the growth of compact clusters, roughly equal in size. However, if the data have a tendency to form long chains, the resulting groupings may be meaningless. Unfortunately, there are no rules available as to which algorithm should be preferred, and this is one of the major pitfalls of hierarchical techniques. Four other hierarchical techniques using different parameters in equation [66] are presented by Jain (1986): the centroid, median and group average algorithms and Ward's method.

Once a dendrogram has been produced for a given set of data (which can be

a problem for large numbers of training samples), a partition is required at some level of dissimilarity which reflects the number of significant clusters occurring in the data. *A priori*, the number of clusters may not be known, and one heuristic used is to "cut" the dendrogram at an appropriate point. In practice, one of the most common methods is to choose a dissimilarity value where there is a large gap in the dendrogram, such as the example shown in figure 6.

As remarked earlier, **Partitional Techniques** have been favoured in the past as the unsupervised classification method for the analysis of clouds in satellite imagery. In partitional clustering, the desired number of clusters m is usually set by the user, based on experience. An initial partition is made by identifying m cluster centres and assigning a number of features to each (sometimes referred to as class kernels). The identification of cluster centres may be done in a number of ways and usually involves some estimate such as choosing a subset of features from composite images (Seddon & Hunt, 1985), interactively identifying some training fields which are felt to be representative of major classes in the image (Pairman & Kittler, 1986), or using an alternative clustering procedure on smoothed and compressed histograms to search for local maxima (Wade *et al.*, 1992). Some authors have preferred to assign initial cluster centres randomly (Desbois *et al.*, 1982; Farki *et al.*, 1992).

The next step is to use a criterion function J to determine how to rearrange the training features assigned to each cluster in a way which converges to the optimum partition of the data. Since the technique therefore allows features to move from one cluster to another, the process is often referred to as *dynamic* clustering. One of the problems with this type of method now becomes apparent. At each iteration, a global criterion function J_G (summed over all cluster kernels) must be found, the value of which is to be minimised. Any relabelling of a feature vector results in a change of the kernels and a recalculation of J_G before any decision can be made as to whether to perform any more relabelling. This leads to severe computational problems in its implementation (for example, the clustering technique of Seddon & Hunt (1985) required 7 minutes of computer time per iteration). Part of the problem stems from the number of different partitions that exist in the given feature space for the training samples used, which was previously discussed in §5.1. Finding the optimal partition from around $m^{N_T}/m!$ different partitions is not computationally feasible.

As a result, most partitional techniques use some method of cutting down the amount of computation, but they then usually give only suboptimal partitions. A popular routine known as the ISODATA algorithm (Ball & Hall, 1967) parameterises each cluster with its current mean value $\tilde{\underline{\mu}}_i$ of the N_i feature vectors in the kernel and uses the squared Euclidean distance metric in the criterion function $J(\tilde{\underline{\mu}}_i)$, similar to that of equation [53]. In this case, the global criterion function

$$J_G = \sum_{i=1}^{m} J(\tilde{\underline{\mu}}_i) \tag{67}$$

$$= \sum_{i=1}^{m} \sum_{j=1}^{N_i} (\underline{\xi}_j - \tilde{\underline{\mu}}_i)^T (\underline{\xi}_j - \tilde{\underline{\mu}}_i) \tag{68}$$

$$= \mathrm{tr}\,(S_W) \tag{69}$$

where the alternative definition $\mathrm{tr}\,(S_W)$ denotes the trace of the within-cluster scatter

matrix (Seddon, 1983; Seddon & Hunt, 1985), derived from scatter matrices which are also used in feature selection (§4.5; Duda & Hart, 1973). Such criterion functions have been used successfully in the cloud clustering schemes of Desbois *et al.* (1982), Seddon & Hunt (1985) and Farki *et al.* (1992). However, as pointed out by Pairman & Kittler (1986), the use of a mean to represent each cluster is restrictive, as it does not take into account the different variances or populations between classes. Such clustering techniques are seen to be the unsupervised versions of minimum-distance-to-means classifiers, discussed in §5.1. In fact, there is a tendency in the literature for authors to use supervised minimum-distance-to-means classifiers at a later stage, when the classes have been identified (Seddon & Hunt, 1985; Farki *et al.*, 1992).

A more appropriate metric would surely incorporate the parametric extension described in §5.1, which is the multivariate normal distribution. In this case, getting rid of the scaling constants from equation [65] results in a new multivariate normal criterion function of the form:

$$J(\underline{\mu}_i, \Sigma_i) = \sum_{j=1}^{N_i} \left[(\underline{\xi}_j - \underline{\mu}_i)^T \Sigma_i^{-1} (\underline{\xi}_j - \underline{\mu}_i) + \ln |\Sigma_i| \right] \tag{70}$$

where $\underline{\mu}_i$ and Σ_i are the model mean vector and covariance matrix for the ith kernel, respectively. Pairman & Kittler (1986) showed that the global criterion function can be minimised optimally when the model mean vector and covariance matrix for each kernel are in fact updated from the mean $\tilde{\underline{\mu}}_i$ and covariance matrix $\tilde{\Sigma}_i$ of the features currently assigned to the kernel. It is as if the maximum likelihood values for the two parameters at any iteration are the current values of those parameters in the global criterion function:

$$J_G(\tilde{\underline{\mu}}_i, \tilde{\Sigma}_i) = \sum_{i=1}^{m} \sum_{j=1}^{N_i} \left[(\underline{\xi}_j - \tilde{\underline{\mu}}_i)^T \tilde{\Sigma}_i^{-1} (\underline{\xi}_j - \tilde{\underline{\mu}}_i) + \ln |\tilde{\Sigma}_i| \right] \tag{71}$$

Now, remembering the Bayesian classification scheme of equation [63], it would perhaps seem appropriate to include the *a priori* class probabilities $P(\omega_i)$ in the metric. This is because the metric in [71] tends to partition the data so as to create clusters of equal size, and can therefore overestimate the size of a low population class (Pairman & Kittler, 1986). Once again, the optimum value of $P(\omega_i)$ can be found at any iteration, and turns out to be simply N_i/N_T, the fraction of training features currently in the kernel of class i. The result is a population weighted Gaussian criterion function which tends to form hyperellipsoidally shaped clusters according to the mean vectors and covariance matrices found in the data:

$$J_G(\tilde{\underline{\mu}}_i, \tilde{\Sigma}_i, N_i) = \sum_{i=1}^{m} \sum_{j=1}^{N_i} \left[(\underline{\xi}_j - \tilde{\underline{\mu}}_i)^T \tilde{\Sigma}_i^{-1} (\underline{\xi}_j - \tilde{\underline{\mu}}_i) + \ln |\tilde{\Sigma}_i| - 2 \ln \frac{N_i}{N_T} \right] \tag{72}$$

If the data conform to the assumptions concerning the Gaussian model, the algorithms will converge to a stable solution, although there is no guarantee that this will be the global minimum. Furthermore, even the local minimum may not be found if the number of features in a cluster becomes too small. To overcome some of these problems, techniques such as "splitting and merging", (which alternately creates

and destroys classes on successive iterations according to some distance thresholds (Seddon, 1983)), and the deletion of sufficiently small clusters (Desbois *et al.*, 1982; Wade *et al.*, 1992) have been used.

After clusters have been obtained from the training data, the question of whether the clusters are significant or merely artifacts of the clustering algorithm should be addressed. Two main areas of cluster validity have been studied (Jain, 1986) and include tests for clustering tendency and tests for the validity of partitions. Clustering tendency refers to whether the data exhibit a predisposition to cluster, and this can usually be tested by comparing the data in feature space against a uniform distribution of randomly generated artificial data. One example is quadrat analysis, where a window in a 2-dimensional feature space is divided into squares of equal size, called quadrats. Under a null hypothesis of uniform data, the number of features in each quadrat follows a Poisson distribution. Unfortunately, this method appears to be inadequate for higher dimensional data.

Tests for validity of partitions seek to find the correct number of clusters in a dataset and whether those clusters are significant. If the basic ISODATA algorithm is used (equation [68]), one possibility is to find out whether the decrease in the criterion function is significant compared with the decrease expected from a random sample of features obtained from a Gaussian distribution. For data assumed to show a multivariate Gaussian distribution, a likelihood ratio test is available for which a chi-squared statistic is expected to fall below some critical threshold (Jain, 1986).

One simple test of clustering that has been performed in cloud detection has been to examine the uniqueness of the classification by comparing adjacent images which have been classified independently, but with the same technique (Desbois *et al.*, 1982; Seddon & Hunt, 1985). The results tend to show that although most samples are found in the same classes, some samples (particularly at the boundaries) are inconsistent. This unfortunately means that each classification is not completely unique, which might for example result in problems in an unsupervised analysis of temporally sequential imagery.

5.3 Introduction to Artificial Neural Networks

At the start of section 3, a quotation from Duda & Hart (1973) stated that although humans are expert at perception, it is not really understood why. Perhaps that statement is a little out of date nowadays: since the publication of the book "Parallel Distributed Processing" in the mid 1980's (Rumelhart *et al.*, 1986a), there has been a great deal of research in modelling the perception capabilities of the human brain through the use of *artificial neural network* models. The use of these models has led to a greater understanding of how humans perceive objects in scenes and imagery and has shown the inherent inability of the architecture of current computers to work in similar ways. The following brief introduction to neural network models attempts to explain what some of these architectural differences are and why neural networks have been considered by many to have excellent potential for improving the classification accuracy of remote sensing data (Lee *et al.*, 1990).

Before looking at the differences, some understanding of the basic unit of the brain is required. These units are called neurons and can be considered to be stan-

dalone analogue processing units. Each one consists of a body called the soma, which has long filaments emanating from it known as dendrites. Dendrites play the rôle of "wire" connectors which bring in information (in the form of voltage pulses) from other neurons. There is one other type of filament emanating from the soma, called an axon. Axons are electrically active and serve as the output for the neuron. At the end of each axon is a synapse which can release chemicals known as neurotransmitters if their potential is raised sufficiently high enough. Synapses can attach themselves loosely to dendrites from other neurons, forming synaptic junctions.

The operation of a single neuron can now be imagined in the following way. Consider a neuron consisting of two dendrites attached to the soma. Each dendrite has one synaptic junction (although in reality there may be more than one). If no neurotransmitters have been released across the synaptic junction, then there is no change in the current state of the soma. However, if neurotransmitters have been released, they may serve to excite or inhibit the dendrite they affect. For the moment, assume they only excite the dendrite. Their excitation, taken *together*, may be enough to "fire" the axon attached at the output of the neuron. There now exists a system such that a number of input excitations taken together can result in a voltage above a certain threshold which in turn fires the output. It is interesting that the brain can modify the coupling between neurons at synaptic junctions by releasing more or fewer neurotransmitters so as to reinforce good connections. This is an important feature of learning in neurons, known as *weighting* in neural network models.

It may now be apparent that the basic structure of the brain has significant differences compared to typical computers. To begin with, there are some 10^{10} neurons, each connected to around 10^4 others (Beale & Jackson, 1992). This implies a massively parallel system where many simple units are each sharing the work, rather than one central processing unit which is performing its processing serially. Such a system is known as a parallel distributed processing system, and has a number of advantages. Contributions made by any one neuron are not too important: if it goes wrong, it is unlikely to affect other neurons significantly (it has fault tolerance), and in any case, with the ability to learn and adapt, the brain can adjust to the loss of a neuron quite easily. If a large number of neurons are lost at any one time, the brain does not just switch off but will behave in a less optimal way. One example is that of deep dyslexia which occurs with certain kinds of brain damage in adults (Rumelhart *et al.*, 1986a). Shown the word PEACH on a card, a subject may say that it reads APRICOT. This is bizarre, because the subject must have accessed the word PEACH in order to make the error! What is interesting though is that it is possible for a parallel distributed network to relearn, even when it has been damaged. The concept of damage resulting in suboptimal performance is known as *graceful degradation.*

It is important to remember that the neural network models considered here are not intended to be detailed copies of real nervous systems; they merely act as models which use a few of the properties of the brain (some of which are exaggerated and some of which are ignored). Furthermore, although the brain has around 10^{10} neurons connected to 10^4 others, model neural networks often consist of only a few tens of "neurons", connected to only a handful of other input or output neurons.

In order to produce a neural network which can model properties of the brain in some way, it would seem obvious to start with a mathematical model of a single

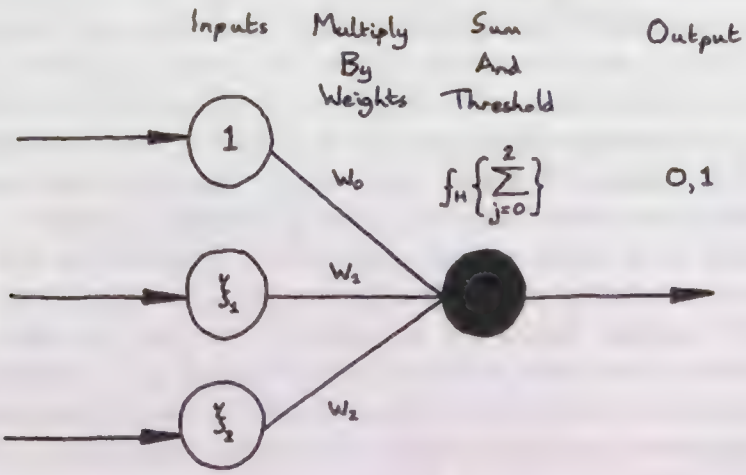

Figure 7: *The components of a perceptron. The 3 inputs in this example ($\xi_0 = 1$) are multiplied by individual weights prior to the main "body" of the model neuron. The body then sums these weighted inputs and thresholds the result according to the Heaviside function, which produces either a "0" or a "1" at the output.*

neuron (see figure 7). The basic function of the biological neuron is to add up its inputs (call them $\xi_1, \xi_2, \ldots, \xi_d$), and to produce an output if the sum is above some threshold value. The efficiency of the synapses can be modelled by multiplying the inputs with weights $w_1, w_2, \ldots, w_d$. Finally, the activity of the neuron a can be obtained by using some thresholding function such as the Heaviside function f_H. If the sum of the weighted input values is above the threshold (call it w_0), then the activity is set to 1, else it remains at 0:

$$a = f_H\left\{\sum_{j=1}^{d}(w_j\xi_j) - w_0\right\}$$

$$\text{with} \quad f_H\{x\} = 1 \quad \text{for } x > 0$$
$$f_H\{x\} = 0 \quad \text{for } x \leq 0 \qquad (73)$$

It is worth comparing this activity function with the discriminant function of equation [57] in §5.1. There, a feature vector $\underline{\xi}$ was assigned to one of two classes, according to whether the discriminant function $f(\underline{\xi})$ was less than or greater than zero. This is identical to equation [73]. Since the inputs $\underline{\xi}$ are passed through the neuron to produce some activity a, the system is referred to in neural network circles as a *feed-forward* model.

How does the basic neuron model learn? In §5.1, a set of training data were used via a gradient descent procedure (equation [59]) and a criterion function, to determine values of the weights iteratively. Training the neuron model is basically the same: some way of reinforcing correct behaviour and discouraging incorrect behaviour is required. One possibility would be to present the neuron with a feature vector as input, calculate the activity through the use of equation [73], and compare it to the

expected activity (so this is supervised learning). The weights are then altered *only* if the result is wrong, by adding or subtracting the appropriate input element for each weight. This, with the addition of a positive gain term ρ (to control the rate of adaption) is the basic perceptron learning algorithm (Beale & Jackson, 1992).

To start the learning procedure, the weights and threshold must be initialised in some way, and this can be done using a random number generator to produce small initial random values. Comparison of the activity a to the known class is performed by coding the class into a 0 if it is class ω_1 (say), and 1 if it is class ω_2. Then, if the activity for some training sample is found to be 1, but it should have been 0, the weights are updated according to:

$$w_j(t+1) = w_j(t) - \rho\xi_j(t) \tag{74}$$

In the opposite case, if the output is found to be 0, but should have been 1, the sign in [74] is reversed. Furthermore, as reinforced learning is required, no change to the weights takes place if the output is as expected.

It is useful to consider this learning algorithm in terms of a 2-dimensional vector. Earlier, in §5.1, it was noted that in 2 dimensions, the discriminant function $f(\underline{\xi})$ is simply a straight line with values of the gradient and intercept dependent on the weight elements w_j. If an initial guess for the weights in the activity function is made, it may be as good as choosing a set of initial random values, and this is like choosing some arbitrary line in the feature space. However, as each training sample is added to teach the neuron, the vector will slowly adjust so as to point in the optimal discriminating direction (somewhat like a weather vane settling down to a constant wind after a blustery period). Figure 8 shows an example of a discriminating function learning the correct classification, after a number of iterations.

Some time after Rosenblatt's original work on the perceptron learning algorithm (1962), Minsky & Papert (1969) decided to query the use of perceptrons in the case of nonlinearly separable problems, which are important for most practical classifiers. In a book entitled "Perceptrons", they showed that basic neuron models could only perform linearly separable problems, and were therefore limited in a similar way to the discriminant functions discussed in §5.1. (One possibility of solving simple nonlinearly separable problems, such as the XOR partition (Beale & Jackson, 1992) is to combine a number of perceptrons into a network. However, some previous knowledge about how to devise the network is required; perceptrons cannot learn this logic by themselves.) This proved to be a major stumbling block for neural network research until a new algorithm was developed by Rumelhart *et al.* (1986b), for use in a network called a *multilayer perceptron.*

With the ability to solve nonlinearly separable problems (as will be shown in §5.4), multilayer perceptrons have proved to become a potentially valuable pattern recognition tool for use in many remote sensing applications (Benediktsson *et al.*, 1990; Lee *et al.*, 1990). This is partly because they are basically data driven learning machines and so do not need any of the model assumptions such as conditional probability densities required for maximum likelihood classifiers. However, they are much more sophisticated than the nonparametric techniques discussed in §5.1, in the sense that they are *nonlinear* nonparametric techniques, for which arbitrary boundaries can be made to separate classes. Some people have in fact ascribed the notion of rub-

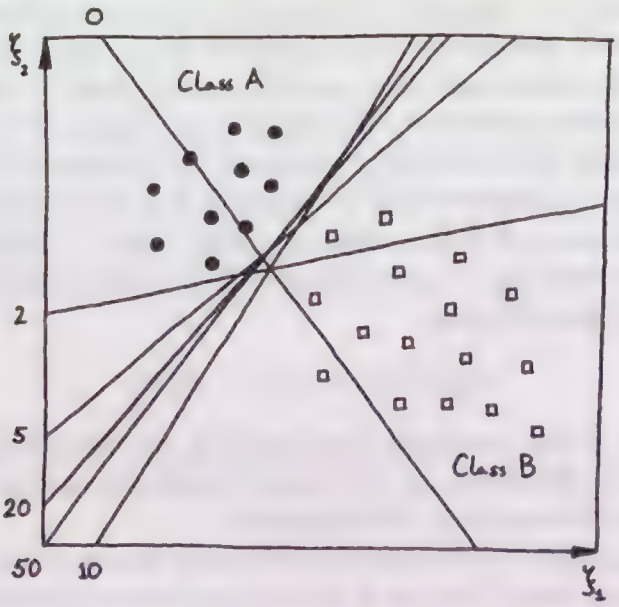

Figure 8: *An example of learning in a perceptron. The activity function is a linear discriminant function, which for each iteration of the perceptron learning algorithm, tends to a position and direction that correctly discriminates the two classes.*

ber sheets to the way in which multilayer perceptrons learn to classify. Whereas the straight line in figure 8 was reoriented in the learning process until the best classifier was achieved for a perceptron, one can think of a rubber sheet bending, expanding and contracting where necessary, to incorporate the training data when teaching a multilayer perceptron. It would appear therefore, that multilayer perceptrons rely purely on the training data given to them in the supervised process. In addition, they have the advantage that once trained, they are computationally more efficient than many of the techniques described in §5.1.

5.4 Supervised Multilayer Perceptron Techniques

One form of basic supervised neural classifier called the perceptron was discussed in the previous section. It was explained that the major drawback of these systems, which effectively put a halt to neural network research in the late 1960's, was the discovery that they were unable to solve nonlinearly separable problems. It was some time later that Rumelhart *et al.* (1986b) showed how it was possible to overcome these difficulties using perceptron networks containing several layers, but with the important addition of two new concepts: *sigmoidal thresholding* and the *backpropagation rule* (sometimes called the generalised delta rule).

One of the problems with the basic perceptron is that the Heaviside threshold function removes information that is needed if a network is to successfully learn. For example, weighted inputs that only just activate a neuron should not be altered to the same extent as those which definitely activate it. This problem is termed *credit assignment*, because a network cannot determine which of the input weights should be increased and which should not.

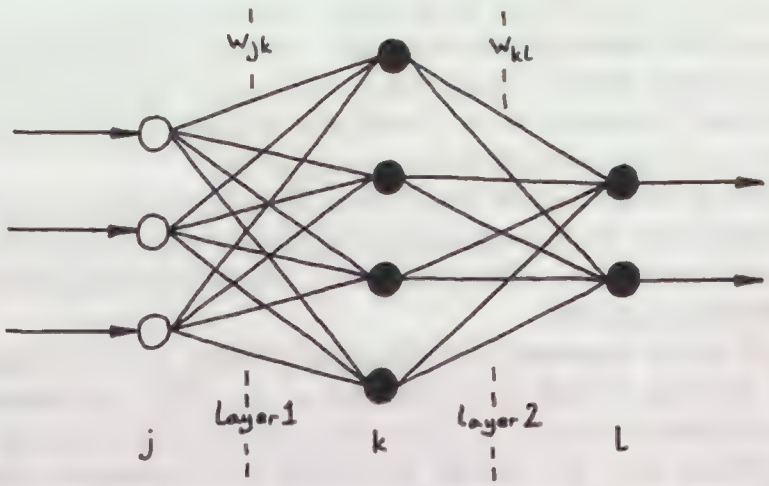

Figure 9: *A simple multilayer perceptron with 3 input nodes, 4 hidden nodes and 2 output nodes. It is best referred to as a 2-layer network (see text).*

A solution is to use a smoothly varying function, such as a sigmoidal function. When the weighted sum has a very large positive value, the output is almost 1. When it has a very large negative value, the output is nearly 0 and in between, the output is somewhere between the two extremes. This results in an output related to its input in a more informative way. In fact, any input-output function which has a bounded derivative will do (Rumelhart *et al.*, 1986b). Another reason for using a nonlinear thresholding function is that any number of layers of linear thresholding perceptrons are no more useful than a suitably chosen single layer (Beale & Jackson, 1992). The result of using a network consisting of a number of layers of nonlinear thresholding perceptrons therefore results in a powerful new tool, the multilayer perceptron.

An example of a multilayer perceptron is shown in figure 9. Here, there are a total of three layers of perceptrons, although it will be shown in a moment that it is better to describe it as a 2-layer network. There are 3 input nodes (forming the input layer), which simply distribute the information on the inputs through to the next layer. The second layer consists of 4 nodes, each of which is just like the basic perceptron, but using a sigmoidal thresholding function. This is termed a *hidden* layer, which as will be seen later, holds *internal representations* of the information flowing through the network. In practice, multilayer perceptrons may consist of more than one hidden layer. The last layer consists of 2 output nodes, each one just like the hidden perceptron nodes. The reason for calling it a 2-layer network is now obvious, as there are 2 layers of perceptrons. Alternatively, one can think of the 2 layers of weights that need to be modified in the network, so that there will be 2 layers of calculations to perform.

In cloud detection algorithms using multilayer perceptrons, different numbers of nodes have been used. The input nodes can be considered to take each element of a feature vector, such as a simple visible grey scale value, the standard deviation of a group of pixels and so on (section 4). Benediktsson *et al.*, 1990), working with Landsat MSS imagery on forest type classes, used Gray coding techniques for the

input, since adjacent integers in Gray code differ by only one digit and are more likely to belong to the same information class. Lee *et al.* (1990) and Lee *et al.* (1992) used 9 and 8 textural features respectively, with a further bias (or threshold) node used in the former. Tovinkere *et al.* (1993) and Walder *et al.* (1994) used a mixture of 8 and 10 spectral and textural features respectively.

The number of hidden nodes required is difficult to assess, but tests have been performed to search for an optimal number beyond which no further improvement is possible (Benediktsson *et al.*, 1990). Typically, the number of hidden nodes in cloud work vary between 20 and 50.

The output nodes represent the classes of interest in the imagery, such as sea, land and low, medium or high cloud (Walder *et al.*, 1994). In training, a value of 1 is assigned as the desired output to one node only (the others are 0), and the aim of training the network is to get this set of outputs to be repeated whenever a similar class sample is presented. Most cloud algorithms use somewhere between 3 and 10 output nodes.

The ability of the multilayer perceptron to solve nonlinearly separable problems does not simply rely on the sigmoidal thresholding function. When a pattern is presented to a network and the activity on the output nodes are obtained, they need to be compared with the desired output, so that the various weights can be adapted as necessary. An error function must be defined which represents the difference between the actual and desired output and which should be reduced in the learning process. This is achieved by calculating the error function at the output, and then propagating this error back through the hidden layers (hence the term "backpropagation").

Consider a single perceptron node somewhere in a network. Assuming that it is at a layer k following layer j, the net input to the node is simply given by:

$$I_k = \sum_j w_{jk} a_j \tag{75}$$

Here, a bias weight w_{0k} may be incorporated into the summation (in the same way as [58]), so that $a_0 = 1$. Notice that the feature elements are relabelled activities a_j, to take into account the previous perceptron node in the network. (For example, for an output node, each input is an output from the previous hidden layer). The new activity from the node at layer k is obtained using the sigmoidal thresholding function (for example):

$$a_k = f_S\{I_k\} = \frac{1}{1 + e^{-\sigma I_k}} \tag{76}$$

where the parameter σ controls the spread of the function. (Large values of σ tend to give the Heaviside function, whilst small values flatten it.)

Let layer k represent the output layer of the network. An error function can be defined at layer k, which is proportional to the square of the difference between the actual (a_k) and desired (o_k) outputs:

$$E = \frac{1}{2} \sum_k (o_k - a_k)^2 \tag{77}$$

This error function must be minimised in some way as part of the learning process. The change in the error function must therefore be measured during training, and

this is easily done by calculating the decrease in the error function E as a function of the change in the net input from the previous layer j. This is called δ_k, and is the reason why the learning rule is sometimes referred to as the "generalised delta rule":

$$\delta_k = -\frac{\partial E}{\partial I_k} \tag{78}$$

The actual method of reducing the error function is with a gradient descent procedure (recall equation [59]), such that on each iteration, the error function E is decreased for a given change in weight w_{jk}. This gradient $\partial E/\partial w_{jk}$ can be expanded using the chain rule:

$$\begin{aligned}
\frac{\partial E}{\partial w_{jk}} &= \frac{\partial E}{\partial I_k}\frac{\partial I_k}{\partial w_{jk}} \\
&= -\delta_k \frac{\partial}{\partial w_{jk}} \sum_l w_{lk} a_l \\
&= -\delta_k \sum_l \frac{\partial w_{lk}}{\partial w_{jk}} a_l \\
&= -\delta_k a_j
\end{aligned} \tag{79}$$

since the derivatives in the summation equal 1 only when $l = j$, and are 0 otherwise. This means that in the gradient descent procedure, a decrease in the error function results in a weight change proportional to the product $\delta_k a_j$. Using a positive gain term ρ in the same way as the procedures in [59] and [74], the gradient descent procedure for weights between two layers j and k (in the forward sense) becomes:

$$w_{jk}(t+1) = w_{jk}(t) + \rho \delta_k a_j \tag{80}$$

(The sign in this update formula is positive, since δ_k was defined as a decrease, but can of course be positive or negative according to the behaviour of each node in the network). The problem now reduces to finding the value of δ at each iteration, for any layer around the network. For any node in the output layer, the value of δ_k is found by using the chain rule on [78], and obtaining the derivatives of [77] and [76]:

$$\begin{aligned}
\delta_k &= -\frac{\partial E}{\partial I_k} \\
&= -\frac{\partial E}{\partial a_k}\frac{\partial a_k}{\partial I_k} \\
&= (o_k - a_k)\frac{\sigma e^{-\sigma I_k}}{(1+e^{-\sigma I_k})^2} \\
&= (o_k - a_k)\sigma a_k^2\left(\frac{1}{a_k} - 1\right) \\
&= \sigma a_k(1-a_k)(o_k - a_k)
\end{aligned} \tag{81}$$

Similarly, for all other nodes at lower layers j, which have to be compared with the node in front at layer k rather than the desired output, one further use of the chain

rule is required:

$$\begin{aligned}
\delta_j &= -\sum_k \left(\frac{\partial E}{\partial I_k}\frac{\partial I_k}{\partial a_j}\right)\frac{\partial a_j}{\partial I_j} \\
&= \sum_k \left(\delta_k \frac{\partial}{\partial a_j}\sum_l w_{lk}a_l\right)\sigma a_j(1-a_j) \\
&= \sum_k \left(\delta_k \sum_l \frac{\partial a_l}{\partial a_j}w_{lk}\right)\sigma a_j(1-a_j) \\
&= \sum_k (\delta_k w_{jk})\,\sigma a_j(1-a_j) \\
&= \sigma a_j(1-a_j)\sum_k (\delta_k w_{jk}) \qquad (82)
\end{aligned}$$

Passing back these error terms through the use of equation [80] is the business of backpropagation. The reason for using the sigmoidal function can now be appreciated, as it has a simple derivative for use in calculating δ.

It is now useful to describe how the multilayer perceptron learning algorithm works with reference to this set of equations. First, all of the weights and thresholds (biases) in the network are set to some small random value, in the same way that they were for the perceptron algorithm (§5.3). A feature vector is obtained for training purposes and is presented to the network in the way discussed earlier (for example, a set of 8 textural features, one to each input node). At the first hidden layer, the net input is computed using equation [75] and the activity is obtained with equation [76]. Each activity is then passed on to the next layer for summation using [75] and so on, until the output nodes are reached. The actual activity and desired activity are now compared at each output node using equation [81] and backpropagation commences. All the weights are now updated using [80] by using the appropriate value of δ (either [81] or [82]). Once all the weights have been updated, a new pattern can be presented, and the process is repeated until either the change in weights drops below a small threshold, or a certain number of iterations have been reached (depending on the size of the training set).

A neural network used for pattern recognition may be viewed as a vector valued function which transforms an input feature vector $\underline{\xi}$ into an output vector $\underline{o}$, representing the class of the feature vector (Lee *et al.*, 1990):

$$\underline{o} = \underline{f}(\underline{\xi}) \qquad (83)$$

In this way, neural networks can be seen as mechanisms for performing computation. The hidden units inside the network act as feature detectors, which learn the mapping in between $\underline{\xi}$ and $\underline{o}$. (A good discussion of the way such a network can solve the XOR problem is given by Beale & Jackson (1992), who show that the hidden units learn to detect when *both* inputs are off or on). This ability for the network to learn the mapping is known as internal representation and has similarities to the way in which the brain appears to store information. It has the capacity to display two of the features of parallel distributed systems discussed in §5.3, namely fault tolerance (the loss of one node would not greatly affect the behaviour of the network) and graceful

degradation (the network would still be able to classify feature vectors with some accuracy, even if a number of weights were scrambled). In fact, introduction of errors into the network can be remedied fairly quickly with a little more training. This is because the solution to the network is found by gradient descent; an error is like a kick out of the global minimum, but it would be quick to reattain the solution back in that minimum with a small amount of learning.

Recall that perceptrons obtain the solution to 2-class problems by estimating the correct linear discriminant function after training. For multi-dimensional feature vectors, this line transforms into a hyperplane. As one hidden layer is introduced into the network (to produce 2-layers), the ability of the network to partition is increased to producing *convex hulls*. These are regions in which any point can be connected to any other by a straight line that does not cross the boundary of the region (such as one of the 4 regions between two intersecting lines, a circle and so on). The reason a network can do this is because the inputs to the hidden layer nodes are effectively lines. The addition of a further hidden layer allows this layer to receive convex hulls as inputs. Combinations of convex hulls can intersect, overlap or separate each other, resulting in arbitrary shapes (thereby producing the "rubber sheets" described in §5.3). This arbitrary complexity means that no more than 2 hidden layers are required (a 3-layer network), a statement known as the Kolmogorov theorem.

Unfortunately, there is one important trap to avoid that has not yet been discussed. This is the possibility for the gradient descent procedure to find a local minimum and stay there, resulting in a suboptimal solution. The error function of equation [77] can be thought of as an energy which varies over a set of 2 weight axes (for example) to produce an *energy landscape*. The training phase tries to reduce this energy by searching for a global minimum. If a local minimum is found instead, then it is as if the solution has become trapped in a well, and will require some extra form of energy to escape to continue the search for the global minimum. A number of approaches exist to avoid such traps, and include gradually lowering the gain term ρ as a function of time (so it has more impact to start), adding internal nodes and adding a momentum term. These two last methods warrant a little further explanation.

Earlier, it was noted that the hidden layer nodes act to form an internal representation of the mapping between the feature vectors and output vectors. The occurrence of local minima amounts to a poor representation within the hidden units (Beale & Jackson, 1992), since two or more classes tend to be categorised in the same way (a good example of this is given in Walder *et al.*, 1994). The simple answer is to add more hidden nodes, thereby improving the internal representation. The other approach is to give the network the momentum required to kick it out of the local minimum (suggested by Rumelhart *et al.*, 1986b). If α is a momentum factor ($0 < \alpha < 1$), then the weight update formula can be rewritten:

$$w_{jk}(t+1) = w_{jk}(t) + \rho\delta_k a_j + \alpha[w_{jk}(t) - w_{jk}(t-1)] \qquad (84)$$

Another type of multilayer perceptron used in the field of remote sensing is that which uses the blocked backpropagation algorithm (Liu & Xiao, 1991). Here, the hidden layers are organised into a number of blocks, each of which contributes to the output layer, rather than all nodes contributing. The results of Liu & Xiao (on Landsat TM data used to classify ground classes over Japan), indicate that the speed

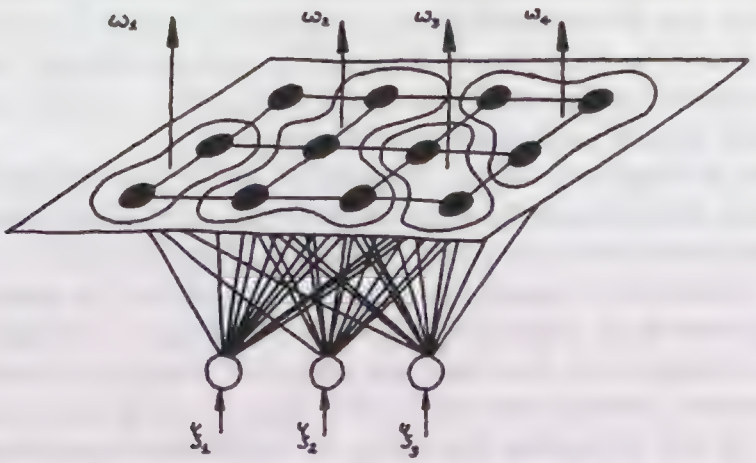

Figure 10: *A Kohonen topological network. The input nodes are connected to every node in the map layer, and the nodes in this layer provide the output for the network.*

of convergence is faster than with normal backpropagation algorithms and that higher classification accuracies can be achieved.

5.5 Unsupervised Kohonen Techniques

Multilayer perceptrons require information about the class to which the input feature vectors belong in order to train the network. An obvious question to ask is: "Does there exist an unsupervised analogue of the multilayer perceptron, for which no predetermined classification is required?" The answer is yes, and the network is named after Kohonen, who had been working with neurobiological concepts such as associative memory, self-organisation and adaptability long before the interest in neural networks was rekindled in the mid 1980's. Neurobiologists had known for some time that localised areas of activity existed in the brain, for example across the cerebral cortex. In addition, certain structures in these localised regions seemed to represent some form of internal mapping of the response from sensory organs (for example, in the auditory cortex), as if they had been self-organised. It was argued that this process must be continuous and that parts of the brain use spatial mapping to model complex structures internally.

To examine this self-organisation, Kohonen introduced a 1-layer, 2-dimensional network, that behaves in the sense of a topological map (figure 10). The neurons are not layered like those of a multilayer perceptron, but are arranged in a grid which displays lateral connectivity. Each input node is connected to every node in the grid, and it is important to realise that the nodes in this grid also form the output nodes. Perhaps the best way to visualise how the Kohonen network learns is to present the algorithm first, and then describe what takes place in the network.

The network is initialised in the same way as usual, by setting the weights w_{jk} between the input nodes j and the grid k to small random values. In addition, an initial value of the *neighbourhood* $N_{k^*}(0)$ of nodes is defined as some large value,

occupying a significant fraction of the output grid. This neighbourhood represents the number of surrounding nodes on the output grid that will be affected as training commences, and its rôle should become obvious in a moment.

The first stage of training is to present a feature vector to the network, with one element per node (let the input layer nodes have subscript j). A minimum Euclidean distance is now calculated between the input and output nodes (layer k), in the same way as equations [56] and [61]:

$$D_K = \min_k \sqrt{\sum_{j=1}^{d} (\xi_j - w_{jk}(t))^2} \qquad \text{for} \quad k = 1, \ldots, K \tag{85}$$

Here, K is the total number of output nodes on the grid and t represents the time (or iteration) when each new training pattern is presented. This results in a minimum distance being found for one particular output node, k^*. The next stage is to update all the weights around this "winning" output node, according to the size of the neighbourhood $N_{k^*}(t)$ at the current time. The weight update formula is simply:

$$w_{jk}(t+1) = w_{jk}(t) + \rho(t)[\xi_j - w_{jk}(t)] \tag{86}$$

where $\rho(t)$ is a decreasing, time dependent positive gain term ($0 < \rho(t) < 1$). Similarly, the neighbourhood decreases in size as a function of time, thereby localising the area of activity around the winning node. Once the weights have been updated, the process is repeated with a new feature vector, until some maximum number of iterations are reached, the decrease in weights has dropped below some threshold, or all the training data have been used.

At each training step, some feature vector results in the weights associated with a particular node and its neighbours being updated. This is called a "winner-takes-all" function, as the winning node claims the input vector and modifies its surrounding weight vectors to align with it. The node has therefore become sensitive to a certain input and will provide a maximum response if that input is presented again: the network is displaying self-organisation. Not only does it become sensitive to the exact input, but since regions are updated instead of individual nodes, any similar input will tend to provide a response at or near this place in the topological map.

It is useful to consider how the self-organisation of the network occurs at the start and end of the training process. At the beginning, the mapping is very coarse, as the network is finding out how many classes the map should eventually contain. The rate of adaption (determined by $\rho(t)$) is deliberately kept large, so that this coarse mapping can occur as quickly as possible. Towards the end of the training process, the map is fine-tuned so that the detail between each of the classes can be represented sufficiently well. One of the problems with the coarse mapping however, is that if the weight vectors were randomly spread, many of them might have a very different orientation to most of the training patterns. They would never come out as winners in the minimum distance algorithm, and so most of the topological map would remain unused. Of course, the optimum distribution would be one where the network was given starting clues as to the number of classes in the training data. Since, in general, this is not known, the technique is to use some form of local neighbourhood around each node to create regions.

In the neighbourhood approach, only a certain number of nodes around the winning node are affected. This means that after just a few feature vectors have been presented, there may still be a few nodes with initial random weights, which can be used by some new feature vector that supplies a very different weight vector. The more grid nodes there are and the smaller the initial neighbourhood, the more likely it is for each class to be distinctly represented. As the neighbourhood is reduced, the effect is for the weights to become specialised toward certain classes or class mixtures. Unfortunately, there appear to be no rules governing the way in which the neighbourhood is reduced, and Beale & Jackson (1992) suggest the best method is to experiment for the particular application. Of course, as in any form of unsupervised classification, the resulting information classes in the output nodes must be identified, and this is why the output grid may be considered as a topological map: classes of interest will be associated with certain places in the output map.

Walder *et al.* (1994) have performed some initial cloud classifications using a standard Kohonen network, searching for 10 different classes (6 of which were cloud classes) in Meteosat imagery taken over central Africa. They used a network of 8^2 output neurons, the results of which indicated that the classification accuracy was only 70% that of a multilayer perceptron. This should not be totally discouraging, since the network was able to distinguish all 10 classes, and ways of improving its accuracy must exist: for example, the use of a larger training set, a larger grid structure and alternative updating algorithms.

Another study using multiply-linked Kohonen networks was conducted by Lewis *et al.* (1992). This method is really a supervised approach, since the (labelled) training feature vectors are presented to one of a number of Kohonens, each used for a particular class. During classification, an unknown feature vector is input to each of the Kohonens, and some decision logic assigns the feature vector to a class based on the closest match between the feature vector and each Kohonen map. In practice, they used 7 textural features and defined 4 classes representing sea, hot land, low cloud and high cloud. Quite a strong unknown class threshold was used (not necessarily a bad thing!), so that only 19% of a Meteosat image taken over south western Europe was classified. The training took 1 hour and the classification of a 500^2 pixel subscene took around 15 minutes, using a PC with an i860 accelerator.

6 Scale Context And Image Segmentation

Now that a number of techniques have been described which can be used to classify samples from satellite imagery individually, it is time to consider how objects identified at different scales can be combined to provide information not just on cloud type, but also cloud systems, synoptic scale features and even meteorological processes. As noted by Peak & Tag (1992) "... identifying a meteorological system is not simply to identify a collection of objects such as clouds, but rather to identify a process".

It may have become apparent in section 4 that different objects might be identified at different spatial scales, using the range of characteristics described there. For example, although nearly all the pattern classifiers for cloud detection described in the literature use spectral and textural features as inputs to identify cloud type, there

is no reason why cloud types themselves (and other attributes such as spatial and contextual features) should not be used on a larger scale to identify synoptic features. (A preliminary study of this type was carried out by Peak & Tag (1992) using only 5 input features to identify 8 classes of synoptic feature; this will be discussed later).

This is where the use of a further pattern recognition tool (that of image segmentation) comes in, to put objects identified at a variety of scales into perspective. It will be shown for example, that segmentation might act as a potential link between separate classifiers, each of which could use different feature vectors as inputs, and operate at different scales to provide a more complete picture.

This section will be used to describe how this type of scale context might result in a clearer understanding of meteorological processes seen in satellite imagery, how image segmentation might be used (with its limitations) and what methods are commonly available. Also, two examples of the use of segmentation in cloud system detection will be described, to give a flavour for the potential applications that will be discussed in more detail in section 7.

Firstly though, what exactly is image segmentation? Nevatia (1986) described it as a tool which can separate components of an image into subsets that correspond to physical objects in the image. The segmented components can then be used by higher-level processes for interpretation and recognition. It is easy to consider image segmentation as a form of pattern recognition, especially when considered from the point of view of human perception. Gestalt psychologists often refer to proximity and similarity measures in the human visual system. In terms of image segmentation, these processes can be considered as the identification of boundaries (differences and edges) and regions (similarities and degrees of homogeneity).

The potential of image segmentation as a tool for cloud system detection is illustrated in figure 11. Here, a large scale region drawn from some visible imagery is shown in the left hand box (on a scale of 1000km). Roughly, the image can be split into three types of synoptic feature: part of a cold frontal system (without vortex) in the centre top, a post-frontal region of open-celled stratocumulus to the left, and some streaks of jet cirrus to the bottom right. It is easy for the human analyst with some satellite imagery interpretation experience to perform this identification, but how should the task be done objectively? One answer is to draw boundaries around the synoptic features in some way, and this is the business of image segmentation.

Additionally, it might be important to identify what is occurring at different scales in figure 11. A trained analyst looking at the left hand box might see the complex area to the centre right, and decide to "blow-up" that portion of the image by a factor of 10 (central box). This is consistent with the view of how humans tend to perceive synoptic features in satellite imagery: they scan the entire image, before going on to locate and concentrate on cloud groupings .(Peak & Tag, 1994). Further location and concentration (the right hand box) might reveal additional detail concerning the processes that are taking place, and this might have a bearing on the general meteorological situation in the image. It would therefore seem that some way of adjusting the analysis from large to small scales and back again might be helpful.

These methods of changing scale to improve image understanding have been categorised into two different techniques: the bottom-up approach, whereby individual pixels might be classified first and synoptic features are identified later (Peak & Tag,

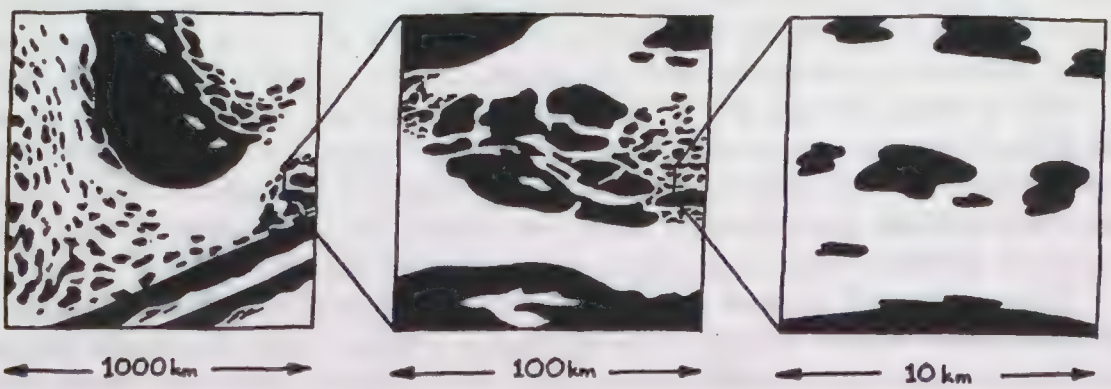

Figure 11: *An illustration of the use of scale context and segmentation in satellite imagery analysis, with cloud shown in black. The central and right hand boxes show different scales taken from the complete box (on the left), and can provide more insight into the meteorological processes occurring in the whole scene. A segmentation of the full image could be performed to identify differing synoptic features, but the use of scale context can improve the segmentation and understanding.*

1992), and the top-down approach which is the reverse and for which image segmentation is an essential starting tool (Peak & Tag, 1994). Now, a third method called bidirectional scale analysis (BSA) is proposed, which is essentially a combination of the two and in which some form of feedback in *both* directions is used to increase or decrease the significance of some particular meteorological phenomenon. The use of BSA should not only improve our understanding of the meteorological situation, but should also increase the accuracies with which regions are classified. A small area which is initially identified as stratus by a cloud classifier might later be recognised as sun glint, in the light of scale context, the meteorological situation and other factors.

In the bottom-up approach, it is of interest to ask what the smallest scale of analysis should be so that sensible cloud classification can be performed. Typically, this will be limited to the resolution of the sensor used, which for the polar orbiting AVHRR instrument is 1.1km at nadir. An interesting study by Welch *et al.* (1989) using high resolution Landsat MSS data showed that textural features calculated at a (degraded) resolution of 228m provided essentially the *same* classification accuracies as those calculated at intermediate scales down to the limiting resolution of only 57m! Accuracies at a lower resolution of 456m were slightly less for stratocumulus and cumulus, but started to deteriorate rapidly for cirrus. At 912m and beyond, classification accuracies decreased for all cloud types, although additional tests with multiple resolution data indicated that some global textural features can take precedence over fine-scale local features. The main result however, is that for any bottom-up analysis of cloud systems, there is no need to obtain a higher resolution than around 500m.

Which of these scale context techniques is chosen depends on the particular application of interest. This in turn will decide what form (if any) of image segmentation needs to be used to link the different scale analyses. For example, if the application is for the detection of synoptic features as a method of improving NWP model

behaviour in data sparse areas (Grant & Bader, 1994), a different type of image segmentation might be required than for the identification of localised cloud type and cover for use in climatological applications. Further discussion of the application of pattern recognition techniques in cloud system detection is left until section 7.

At this point, it is important to realise that image segmentation itself would not provide a very good method for the complete identification of objects in satellite imagery. Synoptic features typically consist of numerous different clouds displaying various grey-levels. Many of the techniques of image segmentation would create physically unreal boundaries in images, because of the tendency of cloud systems to form different shapes, to exhibit different textural properties and to lie adjacent to other different systems (Peak & Tag, 1994). Nevatia (1986) provides a good discussion of some of the inherent problems with regard to texture segmentation.

Image segmentation can be broadly divided into two areas: that of edge and curve detection, and that of region segmentation. The simplest way to detect edges is by calculating gradients in an image. One of the more common methods was described earlier in §4.2 as a method of obtaining textural properties, and is known as Roberts' gradient (an area averaged form of which was given in equation [18]). Another common method is the Sobel edge detector, which uses information from a larger 3^2 neighbourhood. Using the same notation in [18] (where d is just the number of pixels over which the gradient is to be calculated, but with the reference position (m,n) set to $(0,0)$), the Sobel gradient is given by the two components S_x and S_y:

$$\begin{aligned} S_x &= [g(2d,2d) + 2g(2d,d) + g(2d,0)] - [g(0,2d) + 2g(0,d) + g(0,0)] \\ S_y &= [g(0,0) + 2g(d,0) + g(2d,0)] - [g(0,2d) + 2g(d,2d) + g(2d,2d)] \end{aligned} \quad (87)$$

The edges in an image are then obtained by finding the positions of pixels where the magnitude of these types of gradient exceeds some threshold.

Surface fitting is a rather more complicated procedure of edge detection, and relies on the approximation of the intensity profile in an image by an analytical function. The idea is to obtain either the derivatives of this function (the Facet model), or the best fit to a number of ideal edges (with the Hueckel operator to expand the surface into coefficients of a radial Fourier series). A third edge detection approach is known as template masking. Masks of different edge orientation are convolved with the image and a high output is expected wherever significant edges exist. Different sized masks should be used to find the optimal one that gives the highest output (Nevatia, 1986). A fourth technique comes under the heading of second-derivative methods. First derivative methods suffer from the disadvantage that they respond erratically on a ramp profile (where the grey-levels change gradually). However, second-order derivatives are used to locate so-called zero crossings: for a step edge, the second-derivative will be positive on one side and negative on the other.

Once edges have been detected, boundaries are usually obtained by linking or grouping edges. The simplest way to link edges is to connect each edge point to its nearest neighbour, and distance thresholds and orientation can be used additionally to constrain the total number of edges. Grouping can be performed using the Hough transform. Any point in an image can result in a curve in the transformed Hough space (Nevatia, 1986). If a number of these curves cross (at a single point in the Hough space), then there is reason to believe that the points making up these curves

lie on a straight line. The method can be applied to curves of arbitrary shape (rather than straight lines), but with an increase in the dimensionality of the transform space.

The other area of segmentation referred to earlier is region segmentation. Perhaps the simplest approach here is thresholding, where samples occurring in a certain range of some property (such as intensity) are assumed to belong to one group. A good example of the use of this method is character recognition in printed matter, where homogeneous objects typically occur against a high contrast background. The threshold can be selected by experience or by searching for peaks and troughs in a histogram of the property of interest in the image. An extension to this method known as recursive segmentation repeats the process of histogramming, peak selection and thresholding until no new peaks are found, or the regions become too small.

The method of region growing is similar to the process of agglomerative hierarchical clustering described in §5.2. Neighbouring regions are merged according to their relative properties. For example, if two adjacent regions have perimeter lengths P_1 and P_2 and share a common boundary of length L, then they are merged if L/P_1 or L/P_2 exceeds some threshold. The point at which the "correct" segmentation is achieved depends on some measure of the rate of growing, such as the cost of combining regions (Peak & Tag, 1992). An extension to this type of segmentation is known as "split and merge". A tree is used to maintain information about the relationships between regions (such as adjacency or parent-child relations), and the process is switched from merging to splitting or vice versa according to that information.

Nevatia (1986) concludes a descriptive summary of image segmentation techniques by mentioning clustering and classification. Of course, if an image has been classified according to one of these methods, then the resultant thematic map provides a segmentation of the image. However, it may be necessary to use growing or splitting techniques to go on to identify synoptic features (and in addition, further information such as spatial and contextual characteristics may be required to increase the level of confidence in the segmentation).

To close the discussion of scale context and the use of image segmentation, it is perhaps instructive to consider two examples of the use of these concepts in real cloud system detection algorithms. The first example is taken from Peak & Tag (1992), who used a multilayer perceptron neural network (§5.4) to classify a number of synoptic features. They used a meteorological image interpretation expert to identify any "primary" synoptic features in imagery taken over the North Pacific in 1983 from the GOES-West satellite: the result was 172 individual features corresponding to 8 different types. For each feature, the amount of low, high and multilayer cloud was calculated, as well as the longitudinal extent, the latitudinal extent and the northernmost latitude occurring in the feature. Although this may be considered an example of classification only, the interesting aspect is that the products of a cloud classifier and an image segmenter (both human in this case!) were used as inputs to the synoptic feature identifier.

The use of a feature selection technique showed that high cloud was not a very useful feature vector element, and was discarded from later analysis. After training the network on 63 synoptic features, an 83% overall accuracy was obtained for the training samples, and an 83% overall accuracy was obtained on the remaining 109 test samples. The authors considered how this type of classifier could be used in a

bottom-up image analysis. The original pixel data would be classified on a pixel by pixel basis. A segmentation technique might then be used to identify specific regions which could be used as part of the input to a synoptic feature classifier (this part of the process was not explicitly stated in the work, but would seem a logical suggestion given the above discussions).

The second example used image segmentation in a direct initial way to cut up the image into synoptic features (Peak & Tag, 1994). The basic theme was to follow the (top-down) approach of a human analyst described earlier: to scan and segment the image, and then close in and analyse each synoptic feature as required (although this work concentrated on the initial segmentation only). A new scheme called hierarchical threshold segmentation was used to search for the synoptic features. It can be considered a hybrid between threshold and split and merge techniques, but used an artificial neural network to decide on whether to split or merge. Initially, a series of binary images were created at a number of increasing thresholds. Each image therefore contained a number of partitioned objects which would decrease in size as the threshold was increased. Each of these objects was labelled and a tree was constructed which kept the parent-child information at every stage. The idea was then to prune the tree back to splits, and to use a trained neural network to decide whether to keep or prune the splits, so as to keep meaningfully sized cloud features. (The network was trained using a human to prune or keep these splits). Once the tree had been pruned, the regions found at the end of each limb of the tree were used to provide the final image segmentation.

The inputs to the multilayer perceptron network were kept simple: just the size, perimeter length and ratio of these two numbers for each of the parent and largest two children at each split in the tree. Further, a feature selection criterion was used to identify the most useful of these 9 potential network inputs. Experiments were performed using GOES-West and AVHRR imagery (taken around the world), and overall accuracies of 93% and 90% were achieved, respectively. One complex AVHRR image taken over the Arabian Peninsula in 1988 showed a large region of cloud over land, which seemed to show that the method had erroneously performed a segmentation along the coast. However, later analysis showed that the original human segmentation may have been at fault, and that the machine version did in fact show up real cloud edges.

7 Cloud Detection And Other Applications

The most important pattern recognition tools that can be used for the detection and analysis of clouds and cloud systems have now been discussed. It will have become apparent (particularly from section 6) that the exact tools and techniques used in a given pattern recognition system will depend on the application of interest. For example, a Gaussian maximum likelihood classifier with spectral and textural features may be sufficient in a system designed for local climatological studies of cloud type. However, the identification of synoptic scale features for use in NWP analysis and improvement (see, for example, Grant & Bader, 1994), may require cloud classifiers designed for differing scales (perhaps in a bottom-up approach using

neural networks) as well as segmentation techniques to identify significant regions in those classifications.

The task of this section is to look at the different applications of pattern recognition that have been used in the past for cloud detection and related areas (for example, precipitation estimation). Each major area is presented in terms of a brief history of some of the most influential work, to give a flavour of the particular techniques used and the advancements made.

By far the most common application of pattern recognition to meteorological satellite imagery is the accurate classification of specific cloud types to well-defined regions for operational or climatological purposes. (Many authors working on cloud type classification often refer to both areas of application as their purpose of study).

After the launch of the first weather satellites in the early 1960's, manual nephanalyses of imagery were performed by a number of national meteorological services. However, the last archived manual nephanalysis of the world was made available by the National Climatic Centre in 1972 (Harris & Barrett, 1978) and the UK Meteorological Office stopped producing nephanalyses in 1980. This was partly a result of practical reasons and partly due to technological advance, since satellite imagery could by then be transmitted via fax and in some cases digitally in near-real time, helping forecasters to perform their own analyses as required. However, an early study by Young (1967) had shown how much a subjective analysis of cloud cover from satellite imagery could be prone to error, and other factors (such as need for data reduction and cloud wind motion analysis) converged to highlight the potential benefits of using automatic cloud classification schemes operationally. It was also clear that the development of automatic techniques provided the only way of meeting the specification requirements laid down by the International Satellite Cloud Climatology Project (Schiffer & Rossow, 1983).

On the face of it, the easiest parameter to estimate is whether an individual pixel contains cloud or not. However, this proved to be much more involved than expected, and a useful review and discussion of the problem was given by Rossow (1989). The detection of cloud over pixel sized samples is perhaps best approached from a radiative transfer point of view, and typical schemes (such as the APOLLO algorithm of Saunders & Kriebel (1988)) use thresholding techniques to determine whether a pixel is clear or cloudy. Another scheme (Liljas, 1982) used 3-dimensional thresholds or boxes (1 dimension for each of AVHRR channels 1, 2 and 4) to identify 6 different cloud categories, although different sets of classifications had to be developed for different sun elevations. This last method is similar to the parallelepiped classifier discussed in §5.1, and unfortunately suffers from the same set of problems. An improved version of this technique, known as SCANDIA (Karlsson, 1994) is used operationally by the Swedish Meteorological and Hydrological Institute, and is based on thresholding 7-dimensional hyperboxes to determine one of 23 cloud and surface types. In this scheme, the classifier makes use of all 5 AVHRR channels as well as a land mask and the variance in channel 4.

Earlier in §4.1, it was emphasised that such thresholding methods for single pixels really falls outside the scope of this work, as it cannot incorporate aspects of pattern recognition over varying spatial scales (which Rossow often refers to as statistical techniques). The relevance of such techniques was highlighted in the recommen-

dations for future research given by the International Satellite Cloud Climatology Project report (Rossow *et al.*, 1985): "Further study of histogram and other statistical descriptions of cloud morphology on all (accessible) space and time scales could lead to a climatology of dynamics-cloud physics interactions which would contribute significantly to understanding of cloud effects on climate". These statistical techniques formed the basis for many of the pattern recognition studies that were started in the 1970's.

Parikh (1977) gave an account of much of the early pattern recognition work on cloud detection performed between about 1970 and 1976, and made a comparative study of various techniques available at that time for cloud classification. This resulted in the testing of numerous spectral and textural features to characterise regions in the imagery, a decision tree approach for classifier design and four different classifiers. The use of textural features appeared to show little effect on reducing the number of misclassifications, although this later proved to be the opposite case in many similar analyses (Welch *et al.*, 1988). The preferred design parameters resulting from this study included selected combinations of visual and infrared spectral features only, single stage decision trees and maximum likelihood classifiers[5].

A list of some 8 separate applications resulting from cloud classification were provided by Harris & Barrett (1978). These included energy and moisture budget modelling, physical and dynamical meteorological research, climatological research, hydrological studies, agrometeorological programs, oceanographic research, aviation forecasting and other space and research operations, where cloud information may be vital for go or no-go decision making. They also performed a simple linear discriminant analysis to differentiate stratiform, stratocumuliform or mixed and cumuliform cloud from imagery taken from the archive of the Defense Meteorological Satellite Program. They used a 2-dimensional feature space consisting of the standard deviation taken over a 5^2 pixel region and a measurement of vector dispersion (see §4.2). From this basic setup, overall classification accuracies (compared with the results of an experienced nephanalyst) were found to be greater than 72%.

An alternative technique to classify clouds automatically in Meteosat imagery was developed by Desbois *et al.* (1982). They used a minimum-distance-to-means partitional clustering method (see §5.2) to identify significant classes occurring in the imagery in an unsupervised fashion. No features were used other than spectral data from the Meteosat visible, infrared and water vapour channels (almost a pure radiance method, except that the corresponding "thresholds" were obtained by training the classifier in the sense of a pattern recognition technique). The method generally resulted in around 5 or 6 classes including large opaque convective clouds, low cloud, ground or sea, the edges of large convective masses and semi-transparent cirrus. The resultant distribution in feature space was used by the authors to estimate the cloud-top temperature of cirrus in the imagery, since such clouds play an important rôle in the radiation budget, and are often used as wind tracers for high tropospheric levels.

Whereas Desbois *et al.* (1982) assigned their initial cluster centres in a random way, Seddon & Hunt (1985) chose a subset of the (3 spectral) feature space values from a composite image, which was representative of the typical range of radiance values

[5]Although Parikh looked at the design of a cloud classifier, the long-term goal of the project was to segment the resultant thematic map and identify synoptic features for use in wind velocity estimation; this application will be considered again later.

present and which did not substantially affect the resulting partition. They also chose a mechanism for creating and deleting classes as the iteration proceeded (splitting and merging), whereas the work of Desbois *et al.* only allowed for class merging. Unfortunately, the method was found to be particularly time consuming, since the algorithm failed to converge after 20 iterations, and every iteration to find new means and calculate the criterion function for each training sample took 7 minutes. However, low intragroup scattering (as a measure of the clustering performance) was obtained for 3 out of the 4 initial image sections tested, indicating well separated clusters. Many of the cloud systems were found to be represented by more than one cluster after segmentation, which either reflects the layered structure present or changes in cloud density, and more importantly indicates an advantage over the use of supervised techniques, where single cloud types would be more likely to be chosen as the preferred classes. However, the procedure was unable to isolate some clouds which only showed up in one or two of the three wavebands (such as diffuse cirrus).

Part of this latter problem was solved by Pairman & Kittler (1986), who changed the simple minimum-distance-to-means criterion to incorporate variance and class population (equation [72]). Their algorithm was able to detect with greater accuracy classes such as "water with sun glint" and subpixel cumulus. This last category (which was not apparent in the original imagery without careful observation) had originally been labelled low thin cloud by a minimum-distance-to-means classifier.

During the late 1980's, several authors based at the University of Wisconsin concentrated on the detailed use of a supervised Gaussian maximum likelihood classifier (see §5.1) for cloud classification. One of the first was a notable work by Wu *et al.* (1985), which concentrated on the pattern recognition of 6 cloud radiance features and 24 textural features to estimate rainfall rates; this work will be discussed in a little more depth later. Following on, but with the emphasis on studying polar climates, Ebert (1987) considered 30 spectral and 36 textural features (the latter incorporating 6 orthogonal transformations), to classify polar AVHRR scenes into one of 18 cloud and surface classes. After using the 2-class Fisher distance to perform feature selection (equation [54]), 5 spectral and 3 textural features remained. Her work assumed that all features could be described by a Gaussian function, and the classification was performed using a Bayesian decision rule, but with all *a priori* probabilities set equal. A total of 870 samples were used to train the classifier, and the overall accuracy achieved on an independent set of test data was 84%, although manual intervention was used such that if two or more of three expert analysts disagreed, the sample was considered to have been misclassified. This was significantly better than the results of the analysts alone, since all three agreed on only 45% of the cases!

In addition, the classifier was compared against 5 other classifiers, each of which relied on a different set of features, similar to sets that previous authors had used. The results were defined in terms of a percentage skill score S, which is a function of the number of correct classifications N_ω, the total number of classifications N_T and the number of correct classifications expected by random chance N_{random}:

$$S = \frac{N_\omega - N_{\text{random}}}{N_T - N_{\text{random}}} \cdot 100 \qquad (88)$$

As the numbers of categories increase, the skill score approaches the classification

accuracy. Each of the 6 classifiers were used on the original training samples, as well as on a set of artificial samples (100 per class) which contained statistical parameters based on distributions appropriate to each feature set, and synthesised with a Monte Carlo random number generator. The skill score for the 8 feature set (which had previously achieved a skill score of 83% and overall accuracy of 84% on the independent test data) was 92%. In comparison, a 3 feature set identical to that used by Liljas (1982) achieved only 56%. Two other multispectral threshold type classifiers were considered, but the total average skill score of all three remained relatively low at 64%. In addition, two classifiers were produced which used spectral means and standard deviations of two or more channels, similar in a way to the statistical clustering methods of Desbois *et al.* (1982). These achieved an average skill score of 74%, still well below the skill score of the 8 feature set, but a significant improvement on the results of the multispectral threshold techniques alone.

A different type of Gaussian maximum likelihood classifier had been used by Kittler and Pairman (1985), which had incorporated the ideas of varying probability of occurrence in context-conveying neighbourhoods. The general scheme was described earlier in §4.5, and some of the results were noted with respect to the typical contextual probabilities found between different spatial arrangements of cloud type. The classifier also demonstrated an ability to avoid noisy edges, which is a common problem when using standard Bayesian classifiers. These edges were not just a result of some smoothing process however, and any small objects remaining in the classified image were very likely to have arisen from real cloud type variations.

A little after Ebert's (1987) work, another author from Wisconsin (Garand, 1988) performed a similar classification exercise, using 20 cloud classes found in GOES-East imagery taken over the North Atlantic and 13 features chosen for their physical significance rather than on a statistical basis. These included 5 features measuring cloud fraction and height, 2 features concerned with visible reflectance alone, a multilayer index, 3 contextual features and 2 orthogonal transform features. Again, a 2-class Fisher distance was calculated for feature selection. The result was that 2 features were discarded, which were cloud connectivity and the amount of high cloud (this last feature was obtained using equation [1] in §4.1, where cloud height features were discussed in more detail). However, these rejected features were used at a later error-correcting stage of Garand's analysis. Once again, a Gaussian maximum likelihood classifier was used (with equal *a priori* probabilities), although this just formed the first stage of classification. In the second stage, the first choice from the maximum likelihood classifier (class ω_1) was compared against the second choice (class ω_2). The decision rule was to assign samples belonging to class ω_1 to class ω_2 if:

$$\sum_{j=1}^{n} \frac{(\xi_j - \mu_{1j})^2}{\sigma_{1j}^2} > \sum_{j=1}^{n} \frac{(\xi_j - \mu_{2j})^2}{\sigma_{2j}^2} \quad . \tag{89}$$

The sum was performed up to the value n, which could equal 1 or 2, depending on how good the features were at separating any given pair, obtained using the 2-class Fisher distance. This extra stage resulted in an increase to the theoretical class separability of 11%. In a similar way to the study of Ebert (1987), Garand compared the results of an independent test set (1020 samples) to the results of three expert analysts. In this case, the overall accuracy achieved was 79%, whereas if the second-

best choices were considered to be correct (using stage 2), the accuracy increased to 89%. In addition, a comparison of the results was made with statistics of cloud type obtained from ship-based observations taken over 12 years. Many of the cloud classes found with the satellite imagery for the given time period of the data (January and February 1984) occurred with a similar frequency to that of the ship-based observations. Additionally, 41% of the satellite observed cloud systems were found to be multilayered, in comparison to 43% obtained from ship-based climatologies. Significant differences were found however, when the climatologies of the 20 cloud classes were compared for the January and February satellite data. Although the total cloud fraction and mean albedo were practically identical, maps of the cloud fraction and distributions of the cloud classes were quite different.

Textural features were by this stage being seen as having an important part to play in cloud classification algorithms. Welch *et al.* (1988) decided to examine cloud classification accuracies using high spatial resolution textural features alone. Initially, they chose grey-level co-occurrence matrices to define the textural measures (equations [2] to [6] in §4.2) and discovered that a change in the pixel pair distances between 57m (Landsat MSS resolution) and 912m resulted in almost no change in overall classification accuracy, suggesting that the information content of cloud pixels separated by 1km is essentially the same as that at a higher resolution. Chen *et al.* (1989) added to this work by considering sum and difference histogram and grey-level difference approaches (equations [8] to [11]) as an alternative to grey-level co-occurrence matrices for use as the basis of textural measures. The main reason for this analysis was to see whether the use of these simpler and more computationally efficient textural measures changed the overall classification accuracies significantly. It turned out that only a small loss in overall accuracy was achieved using grey-level difference methods, but with accompanying savings in run time and storage space of 39% and 88% respectively. One further result was that reducing the number of grey-levels by as much as a factor of 8 had no significant impact on the overall accuracy of classification. Welch *et al.* (1989) continued their analysis of spatial resolution effects by progressively degrading (averaging) Landsat MSS samples and calculating textural features, rather than changing the textural pixel pair distance. In that study, they used the grey-level difference approach and discovered that up to 228m resolution, practically no loss of classification accuracy occurred for any of stratocumulus, cumulus or cirrus cloud types. At lower resolutions (456m and beyond), the cirrus classification accuracy started to decrease rapidly, followed by stratocumulus and cumulus at 912m. This suggested that very high spatial resolution imagery (such as 30m Landsat TM data) is not required to improve upon cloud classification, and that the *highest* spatial resolution needed is of the order of 250m.

A study by Gu *et al.* (1991) concentrated solely on feature selection for cloud classification, based on a variety of 26 different spectral and textural features, all used in one of AVHRR channels 1 and 4. Channel 3 data were also included, but only 6 simple spectral features were calculated, since noise in that channel might have obscured the validity of the textural features. The total number of features analysed was therefore 58. The 26 different features included the 6 simple spectral features (mean, maximum, minimum, standard deviation, range and ratio of maximum to minimum), an averaged Roberts' gradient (equation [18]), 8 grey-level difference statistics (the

mean and maximum of the mean, contrast, angular second moment and entropy, 7 orthogonal transform features and 4 Moran autocorrelation statistics), each calculated over a different scale length. In assessing the features, both the 2-class Fisher distance and the multiclass transformed divergence were calculated. The results of using the Fisher distance as a feature selection tool indicated that 13 of the 58 features produced the optimum set for class separability, whereas use of the transformed divergence resulted in some of the best features being quite different! This showed that features must be selected using a separability measure appropriate to the intended classification scheme. The other main conclusions were that spectral features were of more value than any of the textural features individually, although textural features could improve upon simple spectral features used alone. In addition, the most effective features in multi-dimensional feature space were those which responded to different scale sizes such as Moran autocorrelation statistics.

The pattern recognition algorithm used by Ebert (1987) was later adapted for use during polar wintertime by using all three infrared AVHRR channels 3, 4 and 5 (Ebert, 1992). Pattern recognition techniques have often been considered as having good potential for polar scene classification, largely because textural features in particular are known to be useful in distinguishing clouds from sea ice and ice sheets in low contrast scenes. It was also recognised that polar processes play an important part in the dynamics of global climate and that cloud cover has a major impact on the amount of radiation reaching the surface. Since major uncertainties exist in cloud climatologies at these latitudes (often as a result of discriminating clouds over snow or ice), the pattern recognition analysis of polar clouds came to receive attention from a number of other workers (Key & Barry, 1989; Key, 1990; Welch *et al.*, 1992; Tovinkere *et al.*, 1993). One of the highlights of Ebert's 1992 study was the identification of two large inland lobes over East Antarctica which appeared homogeneous but showed extremely low brightness temperatures in an AVHRR image taken during the Antarctic winter in 1984. These temperatures (up to 19°C colder than the surface temperature observed at Vostok) suggested the presence of polar stratospheric cloud, the existence of which had only been known for a decade. The use of pattern recognition techniques suggested that it might be possible to obtain the first estimate of a polar stratospheric cloud climatology.

Around 1990, a major change took place in the field of cloud classification, when two papers were published concerning the use of neural network techniques (Key *et al.*, 1989; Lee *et al.*, 1990). As a guide to the potential of neural networks as cloud classifiers, the work by Jonathan Lee *et al.* will be considered here, since they used the same Landsat MSS dataset used by Welch *et al.* (1988) and the same grey-level difference textures discussed by Chen *et al.* (1989). The results showed a marked improvement over classical methods, the authors obtaining an overall accuracy of 93% on an independent set of test data in comparison with an accuracy of 84% using stepwise discriminant analysis (Welch *et al.*, 1989). The architecture used was a feed-forward backpropagation neural network (see §5.4), consisting of 9 input units (one for each of the textural features), 2 hidden layers consisting of 40 nodes and 20 nodes in order, and finally 3 output nodes, one for each of stratocumulus, cumulus and cirrus cloud type. Given that only visible data were used to obtain 9 textural features, the resulting overall accuracy of 93% gave an indication of the results that

might be achieved using other algorithms based on neural networks.

As a guide to forecasting and to provide synoptic cloud cover information over extensive data sparse areas, Victor Lee *et al.* (1992) proposed a similar type of neural network classifier, but based upon 8 grey-level co-occurrence matrix textural features derived from GMS visible and infrared channel data. They trained the network using one of 5 cloud types derived from SYNOP codes reporting synoptic surface observations from about 80 land weather stations. The resulting classification accuracies were much poorer than those of Lee *et al.* (1990), being around 60% and 70% for low and high level cloud respectively. These lower accuracies were attributed to the much weaker resolution of GMS data (1.25km for visible and 5km for infrared), the possible co-existence of other cloud types in the samples used (each measuring 80km by 80km) and potential inconsistencies between the clouds observed from surface observations and the same clouds viewed from space.

The next logical step in the consideration of neural networks as operational cloud classifiers was to compare a number of different network architectures against one another, and also against more traditional approaches, both in terms of classification accuracy and operational speed. Welch *et al.* (1992) began by looking at the results of a stepwise discriminant analysis (SDA) classifier (Welch *et al.*, 1988), a feed-forward backpropagation neural network (FFBP - §5.4) and a probabilistic neural network (PNN). Probabilistic neural networks make use of Bayesian classification, such that class assignment is determined by the largest value of the *a posteriori* class probability density functions (see equation [63]). The initially unknown probability densities are estimated using a set of training samples in a Parzen estimator, a good description of which can be found in Duda & Hart (1973). The PNN consists of an input layer (one input node for each feature), a pattern layer (which produces the dot product of the feature vector and the training pattern of every sample in its defined class), a summation layer and an output layer (to determine the highest value of the probability density function estimates). Clearly, the storage requirement for this type of network is larger than a straightforward FFBP, since every training pattern needs to be stored in the pattern layer.

The results on a set of independent test data formed from one third of a set of 1804 AVHRR polar scene samples gave overall accuracies of 85%, 92% and 89% for the SDA, FFBP and PNN classifiers, respectively. Although the FFBP network was clearly the most accurate, it did take over 13 hours to train in comparison with 90 seconds for SDA and 3 minutes for PNN. However, once trained the FFBP network was by far the quickest method in the testing (or operational) phase, whilst the PNN method proved to be particularly slow (being constructed with 1804 nodes in the pattern layer). Another interesting result from this work was that the 20 best features selected using the Bhattacharyya distance measure (equation [49] in §4.5) from the 218 initially considered, were either simple variants of basic spectral properties, grey-level difference or sum and difference histogram textural measures, or infrared (channel 4) cloud fractions. In fact, none of the run-length or orthogonal transform features were useful, nor were the remaining 8 out of the 9 physical properties which had been based on those of Garand (1988).

Another intercomparison of methods for polar scene identification was performed by Tovinkere *et al.* (1993), except that they made sole use of artificial intelligence

approaches: FFBP, PNN, a hybrid neural network (made of a series of Kohonens (§5.5) followed by an FFBP), a "don't care" perceptron, a "don't care" backpropagation network and a fuzzy logic expert system. The idea of "don't care" networks is relatively new, although it is based upon a concept originally presented by Duda & Hart (1973) (see §5.1). The idea is to reduce the number of linear functions (in the case of the perceptron example) to perform pairwise linear separability only. Instead of calculating $m(m-1)/2$ functions (for m classes) though, only $2m-3$ functions need be determined, and the remainder are "don't care". In the case of the "don't care" backpropagation network, the decision surfaces are nonlinear, but the algorithm means that the hyperplanes do not have to be curved as much to achieve accurate classification. The consequences are substantially less training time and higher classification accuracy.

The same data of Welch *et al.* (1992) were used (with the number of samples expanded to 2218), but only 8 spectral and textural features were employed, being the same as those used by Ebert (1987). Rather than giving the results from a set of independent test data, the theoretical accuracies were calculated from 10 bootstrap samples, each being used with replacement. This meant that the resultant accuracies would be higher than those normally obtained using an independent test set. In the order given above, the overall accuracies were found to be 98%, 97%, 86%, 97%, 99% and 95%, although it was pointed out by the authors that part of the reason for such high accuracy was the use of a special image display system to select "pure" samples from each class. Complex class mixtures were not included either and in fact a general purpose operational classifier was not the intended aim of the work; the task was simply to perform a comparison of various techniques. In this sense, the main result was that backpropagation networks gave the best performance, with the "don't care" extension producing the highest overall accuracy and requiring fewer computational resources.

More recently, a cloud classifier which uses a probabilistic neural network has been designed as input to a satellite image analysis meteorological expert system (Bankert, 1994). The need for this application arose because of the shortage of training time and the frequent personnel rotations experienced by shipboard U.S. Navy forecasters (Peak & Tag, 1992). The cloud classifier is expected to be used either as a stand-alone tool, or as a portion of the input data to a synoptic feature recognition routine to classify cloud patterns (described in Peak & Tag (1994) and in section 6 above). The decision to use a probabilistic neural network was based on the results of the work by Welch *et al.* (1992) referred to earlier. The data were taken by the AVHRR instrument from over 7 maritime regions of the world, in order to provide an assessment of regional influence. Feature selection from amongst 204 candidates was performed in much the same way as in the analysis of Welch *et al.* (1992). A total of 15 features were obtained initially to discriminate 10 classes, consisting of 3 spectral features, 5 grey-level difference statistics, 4 sum and difference histogram statistics, 2 run-length features and 1 physical parameter (visible channel cloud fraction). The PNN was trained on all samples except for one, and was then tested on that sample alone (known as the hold-one out method of obtaining classification accuracy). This procedure was repeated (re-training each time from scratch) for each of the 1633 samples used, and resulted in an overall accuracy of 80%.

To investigate the possibility of a more accurate operational classifier, a 2-level 4-network system of PNNs was designed (similar to the decision trees used by Parikh (1977)), which was expected to show improved classification accuracy. The initial classification was into one of 5 classes defined as low cloud, altostratus, high cloud, precipitating cloud or clear. Subsequent classification at the second level subdivided low cloud into one of stratus, stratocumulus or cumulus, high cloud into one of cirrus, cirrocumulus or cirrostratus and precipitating cloud into nimbostratus or cumulonimbus, giving a total of 10 classes. To estimate the accuracy of the system, the number of samples in each class that were correctly classified in both layers was determined, and was found to be 81%, very similar to the original single level PNN classifier. However, using the top level network that only distinguishes 5 classes resulted in an overall hold-one out accuracy of 91%. This was considered to be acceptable for general classification purposes, whereas for high confidence work, the 10 class PNN was thought to be unsatisfactory. Finally, there appeared to be little difference in class accuracies when data from different maritime regions were added to the original PNN, although unfortunately, there were insufficient samples to train and test a PNN on each region alone.

This last discussion has shown how a cloud classifier might be related to other cloud detection applications; in this case as part of an expert meteorological image analysis system for shipboard forecasting. In the following few paragraphs, some of these other applications are discussed with reference to particular investigations that have been carried out.

A discussion of the potential use of an automated system to classify and analyse extratropical cyclones was presented by Burfeind *et al.* in 1987. Although very simple and rather specific, the pattern analysis method was able to point the way to the possibility of quantitative synoptic analyses as an aid to forecasters. Some of the work of Peak & Tag (1992, 1994) has already been discussed in terms of scale context in section 6, and as mentioned above, their interest was primarily in producing an automated imagery interpretation system for the U.S. Navy. Their first work discussed the possibility of a bottom-up approach to an integrated analysis, and envisaged that a cloud classifier would initially operate at the pixel level. A segmentation of the resulting thematic map would then be passed through another classifier, designed to operate on synoptic scale features. Knowing the location of specific synoptic objects, and with an understanding of their associated conceptual model, an expert system could then be used to provide information on meteorological phenomena, such as the likelihood of clear air turbulence at a particular place. More recently (Peak & Tag, 1994), they have decided to follow a top-down approach, in which the whole image initially undergoes a hierarchical thresholding process (see section 6), following on from which synoptic scale features are identified through a number of neural network classifiers including a basic cloud type classifier such as that of Bankert (1994). From this point, the same synoptic feature interpretation system is used as in the bottom-up approach.

In an effort to improve the Australian National Meteorological Centre mesoscale forecast, Mills & Davidson (1987) used a simple pattern recognition analysis of GMS infrared satellite imagery to enhance the detail of initial moisture fields in otherwise data sparse areas. In their system, dew point depression profiles were assigned to

places of interest by classifying infrared pixels within a 50km radius according to the layer of maximum cloudiness (height), the cloud fraction and the cloud type (either stratiform or cumuliform). The last feature was based on the standard deviation of the cloud-top temperature within the area, with a value greater than 3.5°C indicating cumuliform cloud. If the layer of maximum cloud was below 850hPa, the cloud amount was less than 20% and the cloud type was not cumuliform, than the area was deemed to be "clear". It was pointed out that in some cases, this type of analysis could impact positively on NWP model forecasts (subjectively, it appeared to reflect observed cloud features better than analyses using conventional data), and a suggestion was made that more complex cloud recognition algorithms might be used to refine the cloud recognition criteria.

Under the assumption that a systematic climatological bias is associated with a given cloud class, Garand *et al.* (1989) investigated the usefulness of a cloud classification scheme (Garand, 1988) to estimate those biases and so quantify air surface temperatures and humidity anomalies above the ocean surface. This was performed with a view to using the resultant dew point depression profiles in numerical models in a similar way to that of Mills & Davidson (1987). The results showed that the ability to recognise mesoscale cellular convection patterns was important, since these patterns tend to be associated with large negative deviations of surface air temperature and dew point. Furthermore, the authors indicated that the retrieved profiles should have a significant impact on operational analyses. Garand (1993) pursued the theme of humidity profile retrieval by considering a much simpler set of 4 features (cloud fraction, cloud albedo, cloud-top pressure and water vapour brightness temperature), the last one taken using the Meteosat water vapour channel at 6.7μm. The resulting algorithm provides relative humidity root mean square errors between 13% and 20%, depending on height, which is claimed to be better than current infrared techniques, but slightly inferior to the best microwave techniques.

Another idea for the application of pattern recognition techniques, also with improvement of NWP models in mind, was discussed by Grant & Bader (1994). In this work, they used animated satellite imagery as a basis for developing a conceptual model of the airflow over England on 1st April 1993. The UK Meteorological Office's Limited Area Model had forecast light rain for the 1st April, although in the event, the rain was more widespread and prolonged, and failed to clear the coast until 12 hours later than indicated by the forecast. A re-run of the model was performed using modified cloud motion winds and relative humidities, all of which had been based on the conceptual model derived from satellite imagery. The result agreed better with the observed evolution of the associated depression, and the guidance on the rain area turned out to be as good as the operational forecast run on the morning of the 2nd April. The authors concluded by saying that: "It is possible.to extend this method to other weather systems and has the potential for being used automatically, especially with those for which well-defined conceptual models have been derived".

It was noted earlier that the aim of Parikh's (1977) cloud classification scheme was to identify the same cloud in successive images, and hence estimate cloud motion winds. Since that time, many operational wind generation schemes (automatic and manual) have been derived using sequences of images captured by geostationary satellites (Wade *et al.*, 1992). However, not all cloud motion is representative of the

surrounding wind (e.g. gravity waves, lee waves and banner cloud), and these wind systems have usually been edited by a trained meteorologist. Wade *et al.* (1992) considered the use of a multispectral partitional clustering algorithm (§5.2) similar to that of Pairman & Kittler (1986) to obtain an initial target for tracking automatically. The assumption was that the resulting clusters would extract the principal cloud layers and would provide natural data patterns, similar to those that the meteorologist would have chosen. From a sequence of three visible and infrared Meteosat images, spanning a total of 1 hour, the central image was clustered, and all pixels belonging to a particular class were checked to see whether they provided a population of sufficient size in the target window (greater than 30% filled). Once a suitable target was found, three target tracking algorithms were used to estimate the cloud motion winds, thereby allowing an assessment of each algorithm. The technique was applied to 3 separate sets of Meteosat infrared imagery, and the resulting winds were compared with the UK Meteorological Office's fine resolution model at heights of 850hPa, 500hPa and 250hPa. Although clustered and raw (manually selected) images showed similar mean direction errors and root mean square speed errors of around 3.5° and 6 knots respectively, the clustered images provided many more "valid" wind vectors over the given target area (in the context of goodness of fit to the model winds). The study concluded that not only could an automatic clustering scheme replace inefficient and subjective manual selection, but that clustering before target tracking could significantly improve cloud motion wind estimates, although any advantage might be small for uniform wind fields.

Numerous investigations have been conducted over the years in an attempt to estimate precipitation from satellite imagery, and an excellent summary and intercomparison of recent work undertaken on Meteosat imagery obtained over north west Europe in 1991 has recently been produced (Global Precipitation Climatology Project - Algorithm Intercomparison Project no. 2 (WMO, 1994)). Some of the early techniques relied on subjective cloud type identification (recognition of a variety of cloud patterns) by an experienced meteorologist, and with this in mind, Wu *et al.* (1985) looked at the possibility of applying pattern recognition techniques to the determination of rainfall rates. A Gaussian maximum likelihood classifier was used to classify samples measuring 20km by 20km taken from GOES imagery with the use of 14 different spectral and textural features selected from an original set of 48. The feature selection method used the 2-class Fisher distance, and two separate classifications were performed in sequence: one to distinguish areas with no rain from those with rain, and a second to distinguish rain areas into light (0.5 to 5.0mmh^{-1}) or heavy rain (greater than 5.0mmh^{-1}). To train the classifier, rainfall rates were inferred from National Weather Service radar data, which had been observed simultaneously with the satellite imagery. The rainfall rates R, in mmh^{-1}, were obtained from the measured radar reflectivities Z (mm^{6}m^{-3}), using the relationship:

$$Z = 55R^{1.6} \tag{90}$$

which was in operational use for convective precipitation at that time. Skill scores (using equation [88]) were calculated to establish the overall accuracies of the rainfall rate classification. The result was that for distinguishing rain from no rain using both visible and infrared imagery, an average skill score of 83% was achieved, whilst

for distinguishing all 3 classes, an average skill score of 59% was obtained. Wu *et al.* noted that textural features could improve the skill score by as much as 27% when compared to a feature vector based on spectral features only.

Garand (1989) used his cloud classification scheme (Garand, 1988) to estimate probability of precipitation over oceanic areas, based on the mean probability of precipitation found from ship based rainfall data for each of the 20 cloud classes that might exist. The resulting estimator would therefore classify a sample into one of the 20 cloud classes and assign a likelihood of rain for that class. In addition, a much simpler estimator was considered which only used albedo and cloud-top temperature as direct features. Both methods produced very similar results, except for some specific cloud classes (low stratus and dense cirrostratus), where the maximum difference reached 13% as a result of the lower classification ability using just albedo and cloud-top temperature.

In a more recent study, Grassotti & Garand (1994) used cloud-top pressure, albedo and cloud fraction as satellite based input features to a unsupervised partitional classifier, as well as a 6 hourly estimated model rainfall rate. They trained three classifiers using radar-raingauge composite data from the last 15 days of the first month's set of data taken from the GPCP Algorithm Intercomparison Project no. 1. The classifiers used infrared satellite data only, infrared and visible data, or infrared, visible and model data. The three classifiers were then compared against both model and observed rainfall rates. The main conclusions were that the addition of visible data clearly improved the precipitation estimation compared to the use of infrared data alone, and that the "development of long-term rainfall climatologies on a global scale will certainly require a composite of conventional, forecast model and satellite estimates".

Some authors have considered the use of various pattern recognition techniques with a view to obtaining physical parameters describing cloud fields. Garand & Weinman (1986) used orthogonal transforms to obtain clustering strength in 4 different directions as a means of reproducing a similar image, but using a Markov model of texture to synthesise the clouds. They were not only able to obtain these parameters for a variety of oceanic cloud fields, but effectively produced a data compression factor of the order of one to several hundred.

The idea of using pattern recognition techniques to compress imagery is not new. Liljas (1982), using a multispectral parallelepiped classifier showed that classified satellite imagery could reduce an original set of images covering 3 channels, 2^{10} intensity levels and 2000^2 pixels to a single image containing 2^3 or 2^4 levels and around 500^2 pixels, but retaining the required information for forecasting on cloud type or possibly even precipitation intensity. Such an image could be transmitted via a telephone line within minutes. LeGleau & Derrien (1992) showed how it was possible to provide French forecasters with two AVHRR products that were considered to be of significance, via their Météotel 4-bit network. One product used nighttime AVHRR data and was able to detect low cloud over land or sea and distinguish cirrus from thick cloud, whilst the other was designed to distinguish snow from cloud. By using some basic spectral thresholding techniques, both these products could be sent to display classes into, at most, one of four different colours. The nighttime classification product has been available operationally to French forecasters since 1989.

8 Conclusions

This report has presented the main pattern recognition techniques currently available for spectral, textural, spatial and contextual image characteristics which can be used in a variety of applications to detect and recognise clouds and cloud systems from meteorological satellite imagery. The summary below gives a brief résumé of the main results and techniques described in the sections above. The report is then concluded with a discussion of some of the potential pattern recognition applications that could be used at the UK Meteorological Office together with the recommended lines of research required to achieve those applications.

8.1 Summary

The main emphasis of the report concerning the detection of both clouds and cloud systems (the latter on synoptic scales) in satellite imagery was mentioned early on in the introduction. Pattern recognition techniques as tools for the analysis of such systems were introduced and some of the fundamentals of the subject were highlighted in section 3. These included the route through a typical pattern recognition system incorporating a transducer stage, a feature selection or extraction stage and finally classification. Since many of the aspects of the transducer stage, and in particular the ground processing segment of satellite imagery have been discussed elsewhere (Allam *et al.*, 1990), this report has concentrated on the last two stages of pattern recognition, but has also looked at ideas behind scale context and image segmentation.

The concepts of feature space as a way of collecting together the various image characteristics for pattern recognition were also discussed in section 3. The classification stage was identified as a method of partitioning up this feature space into data patterns, according to whether the training scheme employed is supervised (the classes have already been identified) or unsupervised (in which classes have yet to be identified and are grouped according to more "natural" criteria).

Section 4 presented many different image features which could all be used in a pattern recognition scheme. To provide some form of appreciation of what each feature might be capable of doing, they were split into one of four kinds: spectral, textural, spatial or contextual. In discussing spectral characteristics, it was made clear in §4.1 that pattern recognition studies are not concerned solely with the individual details of radiative transfer models, but are more tied to the relationships between a large range of features which often describe information on a variety of spatial levels (often referred to as "statistical" cloud detection algorithms). Textural features were described in some detail in §4.2 and included a number of important variations including grey-level co-occurrence matrices, grey-level difference statistics, edge detectors, orthogonal transformations, autocorrelation features, mathematical morphology features for tonal primitives and relative extrema densities. Studies which were later discussed in section 7 concluded that grey-level difference statistics and autocorrelation features are perhaps the best available for most cloud recognition work. (Grey-level co-occurrence matrices are also valuable, but use greater computational resources than those of the difference statistics).

Spatial features (which measure large scale spatial structure rather than patterns

which occur frequently in tonal primitives) and contextual features were discussed in sections 4.3 and 4.4 respectively. It became clear that very few authors have used these types of feature in the past, and that they may prove to be of great use, particularly if the scale of recognition is extended to synoptic distances.

In pattern recognition studies, it makes sense to choose a large number of features to begin with, since each application depends on such factors as satellite instrument resolution, the number of classes of interest, whether the training is to be supervised or not, and so on. The candidate features then need some selection so that the ones that provide the best class separability can be used in the final classifier. The aspects of feature selection and extraction were discussed in section 4.5, although it was made clear that selection is the more usual approach. At this stage, the idea of class conditional density functions were introduced, particularly with regard to the commonly assumed multivariate Gaussian probability distribution. In addition, a discussion of maximum likelihood estimates for such distributions was given, because of their importance in the understanding of many of the successful traditional supervised learning techniques. The section then went on to describe a number of probabilistic distance measures, which have been use by many authors to perform feature selection. The most popular have proven to be the 2-class Fisher distance (equation [54]), the transformed divergence (equation [50]) and the Bhattacharyya distance (equation [49]).

A number of pattern classification algorithms were presented in section 5, depending on whether the technique relied on supervised or unsupervised methods, and whether the technique was one of the more traditional approaches or used neural network ideas. Classical supervised methods were discussed first, and considered (in order of complexity) nearest neighbour techniques, parallelepiped classifiers, discriminant functions, and then the parametric minimum-distance-to-means and (Bayesian) Gaussian maximum likelihood methods. Nearly all of these classifiers were implemented by a variety of the schemes described in section 7, where it was shown that the most accurate methods which have stood out in the past were discriminant analyses and maximum likelihood techniques. Some discussion was also given concerning the Gaussian probability assumption so often used in classification schemes: textural features often show far from normal distributions, although some authors have dealt with this problem by splitting classes into subclasses, each represented by its own normal distribution.

In discussing classical unsupervised techniques (§5.2), both hierarchical and partitional methods were considered. Partitional clustering techniques have been favoured in the past in cloud detection algorithms, and have usually relied on criterion functions based on a Euclidean distance metric (the well-known ISODATA algorithm being an example). A more suitable criterion function would appear to be based on population weighted hyperellipsoidally shaped clusters (equation [72]), and this has in fact been used more recently. Questions of cluster and partition validity are important and were mentioned briefly at the end of the section.

With the increase in popularity of neural networks as pattern recognition tools in the late 1980's, such a study could not be complete without some related discussion and comparison with the more traditional techniques. Section 5.3 was designed to give a short introduction to neural network architectures, concentrating on their use

in the field of pattern recognition. The basic perceptron was described in relation to the ideas of discriminant functions discussed earlier in §5.1, to show how through reinforced learning, the perceptron was able to classify linearly separable groups in feature space. The theme of perceptrons was brought up to date with feed-forward backpropagation in §5.4, resulting in multilayer perceptrons, which have already been used in a number of very successful cloud classification projects. Fortunately, an unsupervised analogue of the multilayer perceptron exists in the form of Kohonen style competitive-learning networks. These were discussed in §5.5, with a demonstration of their potential capabilities.

The next major aspect of pattern recognition for cloud system detection to be considered was that of scale context and image segmentation. It was shown that the results of a particular pattern recognition technique used depends on the scale over which feature information is gathered, and that for a cloud system detection scheme, a number of classifiers, each working on a different scale, might be the practical way forward. The ideas of bottom-up and top-down approaches were considered as well as a combination of the two (bidirectional scale analysis), using feedback to decide where to "look next". The desire to work at different scales in this way leads immediately to the tools of image segmentation, and both edge and curve detection and region segmentation were described. Since the combination of these approaches have not yet been used widely in cloud system detection, two examples of recent research were given toward the end of section 6, concerning the design of a satellite image analysis expert system currently being built for the U.S. Navy.

Having described all the various tools of pattern recognition for meteorological satellite imagery analysis, an historical account of the application of cloud classification schemes was given in section 7. This showed how over the last 20 years or so, classification accuracies obtained using independent test data have improved by some 20% or more. Of course, these accuracies depend on the particular application, and hopefully this was made clear during the discussion. One of the important discoveries made in recent years was that a spatial resolution no greater than about 250m is required to extract all of the information relevant to cloud classification; indeed, a spatial resolution of 1km only has a degrading effect of any significance for cirrus cloud. Of particular note was the improvement in classification accuracy with the use of textural features and neural networks. It is envisaged that the use of spatial and contextual features in classifiers which work at different scales will improve the situation further in the near future.

In addition to cloud classification *per se*, other related applications were discussed including the following: the use of cloud system classification and quantitative synoptic analyses by forecasters; input to improve forecasts obtained from NWP models such as the initiation of moisture fields over data-sparse areas and the identification of conceptual models to improve model parameter values; automatic determination of cloud motion winds; estimation of precipitation; physical parameter analysis and satellite image data reduction.

It now only remains to look at some of the potential applications that should be pursued, based on the pattern recognition techniques discussed, and the impact that they might have on the future operations at the UK Meteorological Office.

8.2 Recommendations for Research

Perhaps the main goal of this work has been to identify a pattern recognition technique that will enable the **identification of synoptic cloud systems.** As this research is at an early stage, it is probably a good idea to keep the exact areas of application as open as possible, although it might be useful to develop the technique with a *view* to one particular application, such as the assimilation of conceptual models from satellite imagery into NWP models.

It is envisaged that two parallel studies will initially take place. Both would work in a bottom-up sense, in that features on the smallest scales would be classified first so that the class determinations would be used in some way as subsequent input to higher order synoptic scale classifiers and so on. The justification for choosing this approach is largely based on the results of Peak & Tag (1992, 1994). Although it might seem more natural to work in a top-down way (similar to that of human perception?), some initial segmentation would be required to identify meteorological objects. This process is itself a major study, since recent work has shown that meteorological segmentation must be performed in an intelligent way (Peak & Tag, 1994), and in any case, classification at different scales is still required. (For example, Peak & Tag (1994) aim to use the cloud classifier provided by Bankert (1994)). In contrast, the bottom-up approach has been much better researched, and is well documented, although the process of adding a number of classifiers at different scales will perhaps be more challenging.

Having made these remarks concerning the bottom-up approach, it should be remembered that ideally, the method should be able to change the scale of analysis in *both* directions, depending on the previous results and the use of feedback (the idea of bidirectional scale analysis, discussed in section 6). This means that some of the methods that would be used in a top-down approach, such as segmentation and feedback (in the sense of checking cloud type from synoptic system type for example) need to be kept firmly in mind. Although a BSA scheme will be favoured eventually, for the purposes of the pilot study it would seem more appropriate to use the better understood bottom-up route so as to gain experience and insight into the whole analysis.

In the first study, the principal method would be to use multilayer perceptrons in a supervised fashion, with a large range of features (but concentrating on those which already shown promise in previous work). During the course of the study, feed-forward backpropagation network variants could be considered, especially the "don't care FFBP" described in section 7.

The system would be designed in a sequential way, so that at the first level, a small scale classifier would be used to determine whether cloud exists in a given sample or not, and if it does, what approximate height it exists at, and whether it is multilayered or not. A second classifier would then take the resultant class as output, and add in other features of importance at a higher level (mainly textural). These features could then be calculated over specific regions (produced by segmentation from the previous classifier) to provide cloud type as a class output. A third classifier would then take this data, together with the region information, and perhaps some contextual rules, to determine the synoptic cloud system type, if it exists. The final

result would be a thematic map of cloud system information, but over which extra specific information could be obtained (approximate cloud height, cloud type, cloud fraction). The method would naturally rely on some form of supervised training, but this could be performed at a simple level first (for example, by identifying only 4 or 5 classes per classifier).

Later improvements to the system would make it operationally efficient, perform some form of stringent validation (with the use of ground-truth data, or independent satellite imagery) and examine other features which might increase accuracy and provide a way of increasing the number of output classes (specialisation).

The second (parallel) study would use Kohonen networks in an attempt to perform some form of unsupervised classification of synoptic features. There is the possibility of not only using a sequential system of networks, but also a collection of networks, each working at different scales, in parallel, which would provide output to a final, synergistic network. This approach would be much more experimental than the first, but some of the resultant natural groupings might turn out to be useful to add to the supervised system. In this sense, the Kohonen system would act as an exploratory data analysis tool for the supervised system, and might even prove itself to be the better approach.

Once synoptic scale objects have been identified by the system, matches could be made to conceptual models and relevant parameters extracted for comparison with NWP products. Any significant differences found between the parameters inferred by these conceptual models and those parameters implied in the NWP model data would signal that a model parameter update was required at certain locations. For example, the identification of a particular object might suggest that winds and relative humidities consistent with the appropriate conceptual model should be assimilated into the NWP model, as in the manually adjusted example described by Grant & Bader (1994).

Another area of potentially important research is that of **precipitation estimation** using satellite imagery. This was discussed at some length in section 7, but only for the case of a Gaussian maximum likelihood classifier. Little work has been undertaken to examine the use of neural networks to estimate precipitation. It is likely that many of the features relevant to determining rainfall rates are as yet unknown, so that a Kohonen approach might eventually be productive. However, in this case, a start would need to be made using highly trained datasets to learn what features are of importance. (Lee *et al.* (1990) showed how the summation of weights along various feature-class paths could be used to determine feature significance). This would be achieved with the use of multilayer perceptrons, and the training data would include alternative rainfall sources such as radar and interpolated synoptic weather chart information.

Some work has already been carried out in this area by EDS using experimental semi-supervised Kohonen networks, similar to those discussed by Lewis *et al.* (1992). Initial results with comparisons made against independent radar data showed that the network could achieve a correlation coefficient of around 0.5 between radar rainfall and the satellite imagery rainfall estimate (Holpin, 1994 - personal communication). This simple, first-cut approach demonstrated high potential, since other algorithms achieved correlation coefficients of less than 0.4 in the same study (WMO, 1994).

An interesting application to the detection of synoptic scale cloud systems with the use of pattern recognition techniques is that of **cloud system climatologies.** This would involve a similar design to that described above, with the system running on specific satellite imagery for a long period of time. One of the benefits of using neural networks is that they can learn as time goes on, and it would be very interesting to develop networks containing some form of time information, distributed throughout the weights, so that for example, January climatologies would be kept in the network in a different way to July climatologies. The obvious answer is to include "month" as a feature vector element in its own right. Climatological data would then be held in feature-class paths as discussed above. Obvious products would include cloud fraction over particular areas (totals and distributions) and cloud type and synoptic cloud system distributions.

The ideas contained in this last suggestion puts forward the notion of **cloud pattern research** as a study itself. Kohonen networks, being exploratory data tools, would provide ideal building blocks for learning about a whole variety of features that go to make up familiar cloud structures, and more importantly, for analysing the groupings that tend to occur naturally. This would involve purely data driven research, and the task would be to find out not only what image characteristics corresponded to certain cloud patterns, but also what other physical parameters, obtained from a variety of observations and used as features, were of significance in producing those patterns. The number of potential applications arising from such research might be enormous, and perhaps a neural network system, capable of assimilating any observational data would one day be able to learn to do the job of forecasting.

9 References

Allam R.J., Dibben P., Holpin G., Smith F. & Standley A.P. 1990, "The Use of Satellite Imagery Products for Operational Meteorology", *(Proc. 5th Conference on Satellite Meteorology and Oceanography, London)*, pp.401-403.

Arking A. & Childs J.D. 1985, "Retrieval of Cloud Cover Parameters from Multispectral Satellite Images", *J. Climate Appl. Meteor.*, **24**, pp.322-333.

Ball G.H. & Hall D.J. 1967, "A Clustering Technique for Summarizing Multivariate Data", *Behavioural Science*, **12**, pp.153-155.

Bankert R.L. 1994, "Cloud Classification of AVHRR Imagery in Maritime Regions Using a Probabilistic Neural Network", *J. Appl. Meteor.*, **33**, pp.909-918.

Beale R. & Jackson T. 1992, *Neural Computing: An Introduction*, (Bristol: IOP Publishing).

Benediktsson J.A., Swain P.H. & Ersoy O.K. 1990, "Neural Network Approaches Versus Statistical Methods in Classification of Multisource Remote Sensing Data", *IEEE Transactions on Geoscience and Remote Sensing*, **28(4)**, pp.540-552.

Blum H. 1967, "A Transformation for Extracting New Descriptors of Shape", *Models for the Perception of Speech and Visual Form*, Wathen-Dunn W. (Ed.), (Cambridge,

MA: MIT Press), pp.362-380.

Bunting J.T. & Hardy K.R. 1984, "Cloud Identification and Characterization from Satellites", *Satellite Sensing of a Cloudy Atmosphere*, Henderson-Sellers A. (Ed.), (London: Taylor and Francis), pp.203-240.

Burfeind C.R., Weinman J.A. & Barkstrom B.R. 1987, "A Preliminary Computer Pattern Analysis of Satellite Images of Mature Extratropical Cyclones", *Mon. Wea. Rev.*, **115**, pp.556-563.

Chen D.W., Sengupta S.K. & Welch R.M. 1989, "Cloud Field Classification Based Upon High Spatial Resolution Textural Features 2. Simplified Vector Approaches", *J. Geophys. Res.*, **94(D12)**, pp.14749-14765.

Davis L.S. 1986, "Two-Dimensional Shape Representation", *Handbook of Pattern Recognition and Image Processing*, Young T. & Fu K. (Eds.), (New York: Academic Press), pp.233-245.

Delgado J.A., Casanova J.L. & Illera P. 1992, "Automatic Classification of Clouds Using High and Low Resolution Meteosat Images", *EUM P11 (9th Meteosat Scientific Users' Meeting, Locarno, Switzerland)*, pp.125-130.

Desbois M., Seze G. & Szejwach G. 1982, "Automatic Classification of Clouds on Meteosat Imagery: Application to High-Level Clouds", *J. Appl. Meteor.*, **21**, pp.401-412.

Duda R.O. & Hart P.E. 1973, *Pattern Classification and Scene Analysis*, (New York: John Wiley and Sons).

Ebert E. 1987, "A Pattern Recognition Technique for Distinguishing Surface and Cloud Types in the Polar Regions", *J. Climate Appl. Meteor.*, **26**, pp.1412-1427.

Ebert E. 1992, "Pattern Recognition Analysis of Polar Clouds During Summer and Winter", *Int. J. Remote Sensing*, **13(1)**, pp.97-109.

Farki B., Dagorne D., Guillot B., LeBorgne P. & Marsouin A. 1992, "Classification of Clouds over Africa with Meteosat 4", *Veille Climatique Satellitaire*, **43**, pp.54-77.

Fu K.S. 1986, "Syntactic Pattern Recognition", *Handbook of Pattern Recognition and Image Processing*, Young T. & Fu K. (Eds.), (New York: Academic Press), pp.85-117.

Fukunaga K. 1986, "Statistical Pattern Classification", *Handbook of Pattern Recognition and Image Processing*, Young T. & Fu K. (Eds.), (New York: Academic Press), pp.3-32.

Garand L. 1988, "Automated Recognition of Oceanic Cloud Patterns. Part I: Methodology and Application to Cloud Climatology", *J. Climate*, **1**, pp.20-39.

Garand L. 1989, "Two Automated Methods to Derive Probability of Precipitation Fields over Oceanic Areas from Satellite Imagery", *J. Appl. Meteor.*, **28**, pp.913-924.

Garand L. 1993, "A Pattern Recognition Technique for Retrieving Humidity Profiles from Meteosat or GOES Imagery", *J. Appl. Meteor.*, **32**, pp.1592-1607.

Garand L. & Weinman J.A. 1986, "A Structural-Stochastic Model for the Analysis and Synthesis of Cloud Images", *J. Climate Appl. Meteor.*, **25**, pp.1052-1068.

Garand L., Weinman J.A. & Moeller C.C. 1989, "Automated Recognition of Oceanic Cloud Patterns. Part II: Detection of Air Temperature and Humidity Anomalies Above the Ocean Surface from Satellite Imagery", *J. Climate*, **2**, pp.356-366.

Grant J.R. & Bader M.J. 1994, "Improving NWP Analyses In Data-Sparse Areas", presented at the *Symposium on the Life Cycles of Extratropical Cyclones, Bergen, Norway* (June 1994).

Grassotti C. & Garand L. 1994, "Classification-Based Rainfall Estimation Using Satellite Data and Numerical Forecast Model Fields", *J. Appl. Meteor.*, **33**, pp.159-178.

Gu Z.Q. & Duncan C.N. 1988, "Textural and Spectral Features as an aid to Cloud Classification", *ESA SP-284 (IGARSS '88 Symposium, Edinburgh, Scotland)*, pp.1193-1194.

Gu Z.Q, Duncan C.N., Grant P.M., Cowan C.F.N., Renshaw E. & Mugglestone M.A. 1991, "Textural and Spectral Features as an aid to Cloud Classification", *Int. J. Remote Sensing*, **12(5)**, pp.953-968.

Haralick R.M. 1986, "Statistical Image Texture Analysis", *Handbook of Pattern Recognition and Image Processing*, Young T. & Fu K. (Eds.), (New York: Academic Press), pp.247-279.

Haralick R.M. & Fu K.S. 1983, "Pattern Recognition and Classification", *Manual of Remote Sensing*, 2nd Edition, Colwell R.N. (Ed.), (Falls Church, VA: American Society for Photogrammetry and Remote Sensing), pp.793-805.

Harris R. & Barrett E.C. 1978, "Toward an Objective Nephanalysis", *J. Appl. Meteor.*, **17**, pp.1258-1266.

Jain A.K. 1986, "Cluster Analysis", *Handbook of Pattern Recognition and Image Processing*, Young T. & Fu K. (Eds.), (New York: Academic Press), pp.33-57.

Jones J.G., Thomas R.W. & Earwicker P.G. 1989, "Fractal Properties of Computer-Generated and Natural Geophysical Data", *Computers & Geosciences*, **15(2)**, pp.227-235.

Karlsson K.G. 1994, "Satellite-Estimated Cloudiness from NOAA AVHRR Data in the Nordic Area During 1993", Meteorology & Climatology Report RMK 66, Swedish Meteorological and Hydrological Institute, Norrköping.

Key J. 1990, "Cloud Cover Analysis with Arctic Advanced Very High Resolution Radiometer Data 2. Classification with Spectral and Textural Measures", *J. Geophys. Res.*, **95(D6)**, pp.7661-7675.

Key J. & Barry R.G. 1989, "Cloud Cover Analysis with Arctic AVHRR Data 1. Cloud Detection", *J. Geophys. Res.*, **94(D15)**, pp.18521-18535.

Key J., Maslanik J.A. & Schweiger A.J. 1989, "Classification of Merged AVHRR

and SMMR Arctic Data with Neural Networks", *Photogramm. Eng. Remote Sens.*, **55**, pp.1331-1338.

Kittler J. 1986, "Feature Selection and Extraction", *Handbook of Pattern Recognition and Image Processing*, Young T. & Fu K. (Eds.), (New York: Academic Press), pp.59-83.

Kittler J. & Pairman D. 1985, "Contextual Pattern Recognition Applied to Cloud Detection and Identification", *IEEE Transactions on Geoscience and Remote Sensing*, **GE-23(6)**, pp.855-863.

Lee J., Weger R.C., Sengupta S.K. & Welch R.M. 1990, "A Neural Network Approach to Cloud Classification", *IEEE Transactions on Geoscience and Remote Sensing*, **28(5)**, pp.846-855.

Lee V.C.S., Hung S.L., Cheng A.Y.S., Lam C.Y. & Tam C.M. 1992, "An Operational Cloud Classifier for Satellite Images", *(Proc. 2nd International Conference on East Asia and Western Pacific Meteorology and Climate, Hong Kong)*, pp.480-487.

Leese J.A. & Epstein E.S. 1963, "Application of Two-Dimensional Spectral Analysis to the Quantification of Satellite Cloud Photographs", *J. Appl. Meteor.*, **2**, pp.629-644.

LeGleau H. & Derrien M. 1992, "Combined Use of AVHRR Products and Meteosat Imagery in Nowcasting", *Adv. Space Res.*, **12(7)**, pp.189-192.

Lewis D.J., Corr D.G., Gent C.R. & Sheppard C.P. 1992, "Semi-Supervised Artificial Neural Networks for Classification of Remotely Sensed Images", *(Proc. 18th Annual Conference of the Remote Sensing Society, Dundee, Scotland)*, pp.489-497.

Liljas E. 1982, "Automated Techniques for the Analysis of Satellite Cloud Imagery", *Nowcasting*, Browning K.A. (Ed.), (London: Academic Press), pp.167-176.

Lillesand T.M. & Kiefer R.W. 1987, *Remote Sensing and Image Interpretation*, (New York: John Wiley and Sons).

Liu Z.K. & Xiao J.Y. 1991, "Classification of Remotely-Sensed Image Data Using Artificial Neural Networks", *Int. J. Remote Sensing*, **12(11)**, pp.2433-2438.

Mills G.A. & Davidson N.E. 1987, "Tropospheric Moisture Profiles from Digital IR Satellite Imagery: System Description and Analysis/Forecast Impact", *Aust. Met. Mag.*, **35**, pp.109-118.

Minsky M.L. & Papert S. 1969, *Perceptrons*, (Cambridge, MA: MIT Press).

Nevatia R. 1986, "Image Segmentation", *Handbook of Pattern Recognition and Image Processing*, Young T. & Fu K. (Eds.), (New York: Academic Press), pp.215-231.

Pairman D. & Kittler J. 1986, "Clustering Algorithms for Use With Images of Clouds", *Int. J. Remote Sensing*, **7(7)**, pp.855-866.

Parikh J. 1977, "A Comparative Study of Cloud Classification Techniques", *Remote Sens. Environ.*, **6**, pp.67-81.

Peak J.E. & Tag P.M. 1992, "Toward Automated Interpretation of Satellite Imagery for Navy Shipboard Applications", *Bull. Amer. Meteor. Soc.*, **73(7)**, pp.995-1008.

Peak J.E. & Tag P.M. 1994, "Segmentation of Satellite Imagery using Hierarchical Thresholding and Neural Networks", *J. Appl. Meteor.*, **33**, pp.605-616.

Rosenblatt F. 1962, *Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms*, (Washington DC: Spartan Books).

Rossow W.B. 1989, "Measuring Cloud Properties From Space: A Review", *J. Climate*, **2**, pp.201-213.

Rossow W.B., Mosher F., Kinsella E., Arking A., Desbois M., Harrison E., Minnis P., Ruprecht E., Seze G., Simmer C. & Smith E. 1985, "ISCCP Cloud Algorithm Intercomparison", *J. Climate Appl. Meteor.*, **24(9)**, pp.877-903.

Rumelhart D.E., McClelland J.L. & The PDP Research Group 1986a, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Volume 1: Foundations*, (Cambridge, MA: MIT Bradford Press).

Rumelhart D.E., Hinton G.E. & Williams R.J. 1986b, "Learning Representations by Back-Propagating Errors ", *Nature*, **323**, pp.533-536.

Saunders R.W. & Kriebel K.T. 1988, "An Improved Method for Detecting Clear Sky and Cloudy Radiances from AVHRR Data", *Int. J. Remote Sensing*, **9(1)**, pp.123-150.

Schiffer R.A. & Rossow W.B. 1983, "The International Satellite Cloud Climatology Project (ISCCP): The First Project of the World Climate Research Programme", *Bull. Amer. Meteor. Soc.*, **64(7)**, pp.779-784.

Seddon A.M. 1983, "The Application of Scene Analysis Techniques to Automatic Classification of Atmospheric Data from Multispectral Satellite Imagery", Ph.D. Thesis, University College London.

Seddon A.M. & Hunt G.E. 1985, "Segmentation of Clouds Using Cluster Analysis", *Int. J. Remote Sensing*, **6(5)**, pp.717-731.

Swain P.H. 1986, "Remote Sensing", *Handbook of Pattern Recognition and Image Processing*, Young T. & Fu K. (Eds.), (New York: Academic Press), pp.613-627.

Thomas I.L., Benning V.M. & Ching N.P. 1987, *Classification of Remotely Sensed Images*, (Bristol: Adam Hilger).

Tovinkere V.R., Penaloza M., Logar A., Lee J., Weger R.C., Berendes T.A. & Welch R.M. 1993, "An Intercomparison of Artificial Intelligence Approaches for Polar Scene Identification", *J. Geophys. Res.*, **98(D3)**, pp.5001-5016.

Troup A.J. & Streten N.A. 1972, "Satellite Observed Southern Hemisphere Cloud Vortices in Relationship to Conventional Observation", *J. Appl. Meteor.*, **11**, pp.909-917.

Wade G., Lau K.S. & Wood N.L.H. 1992, "Clustering Applied to Cloud Wind Determination", *Int. J. Remote Sensing*, **13(6)**, pp.3135-3155.

Walder P., MacLaren I. & Reid C. 1994, "Neural Networks and Cloud Classification", presented at the *European Symposium on Satellite Remote Sensing, Rome, Italy* (September 1994).

Welch R.M., Sengupta S.K. & Chen D.W. 1988, "Cloud Field Classification Based Upon High Spatial Resolution Textural Features 1. Gray Level Co-Occurrence Matrix Approach", *J. Geophys. Res.*, **93(D10)**, pp.12663-12681.

Welch R.M., Navar M.S. & Sengupta S.K. 1989, "The Effect of Spatial Resolution Upon Texture-Based Cloud Field Classifications", *J. Geophys. Res.*, **94(D12)**, pp.14767-14781.

Welch R.M., Sengupta S.K., Goroch A.K., Rabindra P., Rangaraj N. & Navar M.S. 1992, "Polar Cloud and Surface Classification Using AVHRR Imagery: An Intercomparison of Methods", *J. Appl. Meteor.*, **31**, pp.405-420.

WMO 1994, *Workshop Report on the GPCP Second Algorithm Intercomparison Project (AIP-2) held in Reading, UK, on 23-26 August 1993*, World Climate Research Programme, (Geneva: WMO).

Wu R., Weinman J. & Chin R. 1985, "Determination of Rainfall Rates from GOES Satellite Images by a Pattern Recognition Technique", *J. Atmos. Oceanic Tech.*, **2**, pp.314-330.

Young M.J. 1967, "Variability in Estimating Total Cloud Cover from Satellite Pictures", *J. Appl. Meteor.*, **6**, pp.573-579.